VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Systém řízení výuky adaptované podle učebního stylu studenta

Learning management system adapted according to student's learning style

2013 Radek Drápela

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Radek Drápela**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Systém řízení výuky adaptované podle učebního stylu studenta

Learning Management System Adapted According to Student's Learning Style

Zásady pro vypracování:

Současné systémy řízení výuky neumožňují měnit předkládanou výukovou látku na základě informací o tom, jaký typ studenta ji právě bude studovat. Existuje teorie adaptivní výuky, ale dosud není realizována v žádném SW systému. Pro její ověření a praktické využívání je nutné vybudovat nový SW systém, ve kterém bude proti existujícím systémům upravená struktura autorské databáze, výrazně zpodrobněna metadata a ralizován zcela nový způsob řízení výuky. Systém musí mít řadu nových funkcí, které rozpoznají učební styl studenta (US), určí mu optimální výukový styl (VS), přidělí mu vhodně upravený výukový materiál odpovídající VS, provází ho výukou, umožňuje průběžné ověřování studentových znalostí a nabízí mu i další informace.

1.Seznamte se s teorií adaptivní výuky.
2.Navrhněte a implementujte základní schéma systému pro řízení výuky se subsystémy Student, Autor, Tutor, Virtuální učitel a Administrátor.
3.Proveďte datovou a funkční analýzu subsystému Administrátor, implementujte jej tak, aby bylo možné dynamicky bez zásahu programátora doplňovat a měnit elementární funkce systému.
4.Proveďte datovou a funkční analýzu subsystému Student a implementujte jeho funkce výukové (s respektováním studentova US a VS), testovací i řešení dalších aktivit.
5.Při návrhu a implementaci spolupracujte s dalšími spoluautory systému, kteří realizují subsystémy Autor, Tutor a Virtuální učitel.
6.Pro řízení celého systému a pro uvedené 2 subsystémy zpracujte uživatelskou a programátorskou příručku.

Seznam doporučené odborné literatury:

Kostolányová K.: Teorie adaptivní výuky.
Dále dle pokynů vedoucí diplomové práce.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **doc. RNDr. Jana Šarmanová, CSc.**

Datum zadání: 16.11.2012

Datum odevzdání: 07.05.2013

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.*

V Ostravě 29. dubna 2013 ..... Radek Drapela .....

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 29. dubna 2013 ..... Radek Drapela .....

Velmi rád bych tímto poděkoval paní doc. RNDr. Janě Šarmanové, CSc. za profesionální a důkladný přístup k vedení vývoje celého projektu, za metodické nápady při návrhu a implementaci, za ochotnou pomoc během celého dvouletého období vývoje systému, při kterém jsem využil mnoho cenných rad a získal mnoho praktických zkušeností, a za to, že mě začlenila do vývojového týmu. Dále bych poděkoval paní Ing. Kateřině Kostolányové, Ph.D., která, jakožto autorka teorie adaptivního e-learningu, pomohla nasměrovat vývoj systému správným směrem. Nakonec bych také velmi rád poděkoval spolupracujícím kolegům, se kterými jsme tvořili zdatný a úspěšný vývojový tým, jmenovitě Ing. Emilii Šeptákové, Mgr. Markétě Dvořáčkové, Ing. Ondřeji Takácsovi a Ing. Michalu Burdovi, Ph.D.

Abstrakt

Tato diplomová práce se zabývá vývojem softwarové webové e-learningové aplikace pojmenované Barborka 4. Jedná se o systém řízení výuky, vytvořený v programovacím jazyce PHP, vycházející z frameworku Nette, používající databázi MySQL, a založený na teorii adaptivního e-learningu. Dle této teorie je studentům předkládána výuka na míru, která se adaptivně přizpůsobuje a mění podle individuálního učebního stylu každého studenta. Adaptivní mechanismus usnadňuje studentům pochopení výukových materiálů rychleji a efektivněji a doposud nebyl v žádném softwarovém systému implementován. Proto bylo potřeba tuto teorii ověřit a vybudovat nový systém pro praktické použití. Součástí systému jsou také funkcionality spojené s výukou, jako je samotné testování studentů, odevzdávání úkolů či projektů, ale také funkcionality spojené s administrací, jako je správa studentů, pedagogů, předmětů, výukových materiálů a jiných. Na vývoji se podílelo více spolupracujících programátorů, kdy se každý věnoval své definované části. V tomto případě byl základním cílem návrh a implementace třech hlavních částí systému: administrátorského subsystému, studentského subsystému a řídicího systému. Diplomová práce obsahuje postupně hlavní kroky vývoje tohoto systému, což jsou specifikace požadavků, vycházející ze zadání a vytvořených prototypů, datová a funkční analýza, návrh uvedených subsystémů a jejich implementace.

Klíčová slova: adaptivní výuka, e-learning, systém řízení výuky, Barborka 4, učební styl, vývoj systému, subsystém, student, administrátor, analýza, návrh, případ užití, implementace, Nette, MVC

Abstract

In this Thesis the development of web-based e-learning software application named Barborka 4 is presented. It was coded in PHP programming language, built up on Nette framework, and it is using MySQL database. Barborka 4 is a learning management system based on the theory of adaptive e-learning. Made-to-measure learning materials are presented to the students in accordance with the theory. The materials are adaptively personalized and changed according to the individual learning style of each student. Adaptive mechanism facilitates understanding the learning materials quickly and efficiently and it also has not been implemented in any software system yet. Therefore, this theory had to be tested and a new system for practical use had to be built. Other system functionalities related both to learning (the student testing, tasks or project submitting etc.) and to the administration (administration of students, teachers, subjects, educational materials etc.) are integral parts of the system. More cooperating programmers were involved in the project with well defined part of the problem assigned to each one of them. As for this Thesis, the fundamental aim was to design and implement three main parts of this system: administration subsystem, student subsystem and control system. The main steps of the system development are presented in this Thesis namely the specification of requirements that arose from the system definition and from the created prototypes, data and functional analysis, the design and implementation of the subsystems.

Keywords: adaptive learning, e-learning, learning management system, Barborka 4, learning style, system development, subsystem, student, administrator, analysis, design, use case, implementation, Nette, MVC

Seznam použitých zkratek a symbolů

AJAX	–	Asynchronous JavaScript and XML
AVS	–	aktuální výukový styl
CRUD	–	Create, Read, Update, Delete - databázové operace
CSS	–	Cascading Style Sheets - kaskádové styly
CSV	–	Comma-Separated Values - souborový formát
DFD	–	Data Flow Diagram - diagram datových toků
DI	–	Dependency Injection - vkládání závislostí
ERD	–	Entity-Relationship Diagram - diagram vztahů databázových entit
ESF	–	Evropský sociální fond
HTML	–	HyperText Markup Language - značkovací jazyk pro tvorbu webových stránek
ICT	–	Information and Communication Technologies - informační a komunikační technologie
IDE	–	Integrated development environment - vývojové prostředí
LDAP	–	Lightweight Directory Access Protocol - protokol pro ukládání a přístup k datům na adresářovém serveru
LMS	–	Learning Management System - systém řízení výuky
MVC	–	Model-View-Controller - softwarová architektura
OOP	–	Objektově orientované programování
OSU PdF	–	Ostravská univerzita v Ostravě, Pedagogická fakulta
OVS	–	osobní výukový styl
PDF	–	Portable Document Format - souborový formát
PHP	–	PHP: Hypertext Preprocessor - skriptovací jazyk
SQL	–	Structured Query Language - dotazovací jazyk
SVN	–	Apache Subversion - systém pro správu a verzování zdrojových kódů
UC	–	Use Case - případ užití
US	–	učební styl
VŠB-TUO FEI	–	Vysoká škola báňská - Technická univerzita Ostrava, Fakulta elektrotechniky a informatiky

Obsah

Seznam tabulek

Seznam obrázků

1 Úvod

V posledních letech, kdy došlo k velkému rozvoji a využití počítačů ve školském i osobním životě, se středoškolská a vysokoškolská výuka začala postupně digitalizovat. U mnoha předmětů došlo k nahrazení tištěných výukových skript a opor elektronickými materiály, dostupnými vesměs i na internetu. Počítače tímto zrychlily a usnadnily výuku. Kromě toho se začaly objevovat pokusy o vytvoření počítačového software, jenž by mohl nahradit samotné pedagogy, uměl by adaptivně vyučovat studenty a řídit výuku sám. Výsledkem těchto pokusů byl vznik několika teorií, které často nebyly přivedeny do praxe formou konkrétního a funkčního adaptivního e-learningoveho systému [1].

Nově vyvíjený e-learningový systém Barborka 4 ovšem vychází z teorie, jež je vytvořená hlavně doc. RNDr. Janou Šarmanovou, CSc. a Ing. Kateřinou Kostolányovou, Ph.D. Systém zprostředkovává adaptaci výuky podle aktuálního učebního stylu studenta. To znamená, že je schopen vytvořit výukové materiály studentovi na míru. Každý student má unikátní vlastnosti, které se promítají do jeho tempa i stylu učení, a tak výuka většího počtu různých studentů je neefektivní. Proto se teorie zaobírá individualizací výuky pro každého studenta zvlášť. Tomu napomáhá i fakt, že počítače jsou určeny hlavně pro jednotlivce. Projekt Barborka si klade za cíl prakticky ověřit tuto e-learningovou teorii během ostrého provozu.

Barborka 4 je modulární systém, rozdělený na subsystémy podle typů uživatelů, kteří se systémem pracují, a to jsou studenti, tutoři, autoři, experti a administrátoři. Systém Barborka 4 obsahuje i takové funkce, které nejsou spjaty s teorií adaptivní výuky. Jedná se vesměs o různé školské administrační a tutorské funkcionality automatizující pedagogické činnosti, jiné aktivity studentů během samotného studia či různé evidence školských údajů. Ve výuce se vyskytují multimediální výukové komponenty, jež jsou znovupoužitelné i během testování či u pokynů úkolů, což usnadňuje autorům opor práci. I přestože je pro autory náročnější výukové materiály vytvořit, investovaný čas se v budoucnu vrátí nejen podobě mnoha učitelských činností, které systém bude automatizovaně zvládat, ale také v podobě lepších výsledků studentů.

Cílem této diplomové práce byl podle zadání především vývoj řídícího systému, studentského a administrátorského subsystému. Ty tvoří základ celého systému Barborka 4. Na začátku je popsaná samotná teorie, hlavně části, které se projevují do následné implementace. Samotný vývoj je prováděn od září roku 2011 až do současnosti týmem, který je složen vysokoškolských pedagogů, metodiků a programátorů.

Pro uvedené tři části systému bylo potřeba definovat všechny funkční požadavky s vytvořením hierarchie elementárních funkcí včetně popisu, vstupních dat a výstupních sestav. Většinu požadavků bylo potřeba získat během vývoje diskuzemi a vytvářením prototypů, ale některé byly definovány přímo, a ty jsou detailněji popsány. Součástí jsou i nefunkční požadavky na samotný vývoj a vlastnosti systému. Nejdůležitější požadavek byl vytvořen pro samotnou strukturu systému, která se odvíjí od definované hierarchie funkcí, podle které se generují všechny části menu a provádí se celá logika oprávnění. Pro tuto strukturu byla požadována potřeba ji administrátorem dynamicky měnit bez samotného zásahu programátora.

Dále byla provedena datová analýza, obsahující definici a popis databázové struktury pro všechny definované funkce včetně vazeb jednotlivých databázových tabulek. Poté následovala funkční analýza, obsahující grafy datových toků a popis případů užití elementárních funkcí. V této práci se nachází jen vybrané případy užití, které jsou pro systém a jeho funkcionalitu důležité. Poté byl proveden návrh systému, jehož součástí je hlavně návrh celé struktury systému, indexové analýzy a seznam s popisem všech použitých tříd včetně třídních diagramů. Nakonec byla navržená funkčnost naimplementována a otestována.

Na závěr byla vytvořena uživatelská a programátorská příručka. Příručka uživatele popisuje grafické rozhraní, všechny dostupné funkce a usnadňuje orientaci v systému. Příručka pro ostatní programátory obsahuje popis pravidel pro vývoj a strukturu systému. Dále obsahuje popis použití vytvořených komponent a dostupných metod.

2 Individualizovaná výuka a e-learning

Se zlepšujícími se technologiemi každým rokem přibývají nároky na vzdělávání a výuku v elektronické podobě s využitím informačních technologií a s podporou moderních metod. Současné počítače a ostatní elektronická zařízení nabízejí prostředky ke studiu, které v minulosti nebyly dostupné. Zobrazují nejen text, ale i animace, zvuk či video, a umožňují poskytovat studentům mnohem konkrétnější a přesnější informace. Při klasické výuce studenti nemohou vstřebávat látku každý svým vlastním tempem. Učitel přednáší skupině studentů danou látku ze stejného učebního materiálu a každý student má jiný učební styl i jiné nároky na studium. Někteří studenti potřebují na pochopení látky více času, jiní čekají na ostatní.

Klasická skripta jsou sestavena podle klasických zásad a pedagogických modelů. Zprostředkovávají informace pouze statickým a neosobním způsobem. Také v nich chybí multimediální komponenty, které mohou plnit funkci motivačního prvku. Na druhou stranu je e-learningový kurz dynamický, vyvolává neustálou aktivitu studenta se systémem, plně využívajícím informačních technologií. Oproti klasické výuce se také dokáže zaměřit na individuální vlastnosti či potřeby každého studenta a automaticky upravit výukové materiály každému studentovi na míru. Tato forma výuky je ovšem prakticky dosud nerealizovaná v žádném systému řízení výuky (LMS).

E-learning je vzdělávání za podpory počítačů (computer based training), které poskytuje informace, školící materiály a vzdělávací obsah pomocí různých forem elektronických médií [2]. E-learning je obecněji definován jako výuka s využitím výpočetní techniky a internetu [3]. Dále mohou být e-learningem označeny i různé úrovně počítačem podporované výuky, kde na nejjednodušší úrovni může být prostá učebnice v PDF dokumentu a na nejsložitější právě LMS [4]. V případě systému Barborka je e-learningem nazýváno využití internetového prostředí pro výuku spolu s řídícím výukovým systémem, který je schopen vykovávat automatizovatelné funkce spojené s procesem výuky [5].

Výhodou adaptivního e-learningu ohlížejícího se na individuální učební vlastnosti a potřeby studenta je rychlejší a efektivnější proces učení studenta než při běžné neadaptivní výuce. Student tedy danou látku mnohem rychleji a efektivněji dokáže pochopit a získat tak nové znalosti přirozenější cestou.

2.1 Činnosti v rámci e-learningu

Typické činnosti prováděné v LMS Barborka 4 v rámci e-learningu jsou:

Poskytování vzdělávacího obsahu
Systém slouží jako úložiště všech výukových materiálů. Mezi tyto materiály patří dílčí výukové texty, dokumenty, multimediální soubory, Java aplikace či flashové animace. Výukové materiály umí systém zpracovávat z disku či z databáze a sofistikovaně zobrazovat.

Studium výukových materiálů
Každému studentovi v systému je na internetu umožněno studium dostupných výuko-

vých materiálu. To znamená v nepřetržitém provozu 24 hodin denně u všech předmětů studenta. Mezi dostupné se řadí předměty veřejné a dále studentem zapsané v daném akademickém roce. Systém provází studenty výukou a také průběžně ověřuje jejich nabyté znalosti pomocí testovacích otázek.

Sebetestování
Po vykonání výuky se studenti můžou pro ověření svých znalostí otestovat na takzvaných autotestech. Jedná se o testy nanečisto, které můžou provádět několikrát za sebou bez omezení do určitého termínu. Výsledky nemají žádný vliv na hodnocení. Autotesty mají stejnou strukturu a stejné vyhodnocení jako testy ostré, takže vykovávání autotestů můžou studenti chápat i jako přípravu k těmto testům.

Ostré testování
Pro zkoušení a ověřování vědomostí studentů slouží ostré testování. Jedná se zpravidla o testy, které jsou součástí zápočtů a zkoušek. Výsledky tedy mají vliv na celkové hodnocení. Studenti se na ostré testy přihlašují a mohou vykonat pouze daný počet testů podle počtu vypsaných termínů.

Vyhodnocení testů
Autotesty i ostré testy se automaticky systémem vyhodnocují a výsledky jsou ihned po ukončení zobrazeny studentům. Výsledky autotestů studenti vidí přímo po ukončení. Poté už nemají možnost si výsledky zobrazit. Jinak je tomu u ostrých testů, kdy si studenti můžou vyhodnocení zobrazovat kdykoli v průběhu daného termínu.

Tutor si může v průběhu testování zobrazit aktuální vyplněné odpovědi studentů i s hodnocením. Po ukončení může zpětně manuálně zasahovat do vyhodnocení a vyhovět tak opodstatněným stížnostem či upřesnit automatické bodové hodnocení.

Odevzdávání projektů
Jako poslední činnost, kterou systém umožňuje studentům vykonávat v rámci e-learningu, je odevzdávání projektů. Studenti mohou do určitého termínu vkládat do systému své textové práce či projekty jako soubory. Součástí odevzdávání je i diskuze mezi studentem a tutorem předmětu, která může obsahovat různé dotazy a odpovědi k projektům, zhodnocení projektů či možné další nahrané verze.

2.2 Základní terminologie

Pro správné pochopení obsahu je třeba vysvětlit a objasnit převzaté použité pojmy a základní terminologii adaptivního e-learningu 1 od autorky teorie adaptivního e-learningu Ing. Kateřiny Kostolányové, Ph.D. [1].

Termín	Vysvětlení
Student	Osoba libovolného věku, která se učí, studuje; je hlavním objektem procesu vzdělávání. V tomto případě se jedná především o studenty vysokoškolské a středoškolské, studující distanční nebo kombinovanou formu prostřednictvím e-learningu.
Virtuální učitel	Řídící interaktivní výukový program, který je schopen řídit výuku studenta (s využitím výukových opor) podobně jako učitel živý. V tomto případě virtuální učitel řídí výuku podle učebních charakteristik studenta.
Učitel – autor	Učitel, který tvoří (obvykle s řadou specialistů) výukové opory.
Učitel – tutor	Učitel provázející studenta distanční formou výuky; neučí, ale zprostředkovává a organizuje výuku, komunikuje se studenty, vyhodnocuje výsledky.
Předmět	Výukový předmět jednoho semestru (VŠ) nebo školního roku (SŠ).
Výuková opora	Soubor studijních materiálů vytvořených pro předmět, zahrnující obvykle textovou učebnici (často dělenou do dílčích částí) a případně multimediální prvky mnoha typů, jako audionahrávky, videonahrávky, animace, interaktivní výukové programy apod.
Učení	Činnost studenta vedoucí k získání jeho nových znalostí, dovedností a postojů. V tomto případě se jedná o e-learningové učení.
Vyučování	Činnost učitele vedoucí k tomu, aby studenti získali potřebné znalosti a dovednosti.
Výuka	Proces učení a vyučování, řízený učitelem, živým nebo virtuálním.
Distanční výuka	Výuka, při níž jsou student a učitel – tutor převážně nebo trvale fyzicky odděleni.
E-learningová výuka	Distanční výuka realizovaná prostřednictví informačních a komunikačních technologií (ICT).
Individuální výuka	Výuka zaměřená na jedince.
Personalizovaná výuka	Individuální výuka přizpůsobitelná na míru studentovi obsahem, formou, způsobem vedení apod.
Adaptace výuky	Personalizovaná výuka s přizpůsobením obsahu a formy výukové opory volbou vyučovacích metod a vyučovacích stylů dle charakteristických vlastností a znalostí studentů.
Učební styl	Postup učení, který student používá v určitém období života ve většině situací pedagogického typu.
Výukový styl	Soubor vyučovacích metod, které učitel uplatňuje ve výuce.
Osobní výukový styl	Postup výuky, který zohledňuje individuální charakteristiky studenta.
Aktuální výukový styl	Aktuální postup výuky, který zohledňuje osobní výukový styl studenta a je sestaven z výukových opor, které má virtuální učitel v danou dobu k dispozici.

Tabulka 1: Základní terminologie

2.3 Learning Management System

Learning Management System (zkráceně LMS) je automatizovaný systém zaměřený na řízení výuky, ve kterém jsou kromě činností e-learningu implementovány jiné důležité moduly a funkce, spadající do běžné potřeby škol či univerzit. To jsou funkce vesměs administrační. Jedná se o správu všech škol, fakult, kateder, studijních plánů, oborů, vyučovaných předmětů, výukových materiálů a o správu uživatelů systému (tj. zaměstnanců a studentů). Dále se mezi tyto funkce řadí i správa anket či dotazníků a kompletní správa všech souborů, které systém obsahuje a se kterými pracuje.

Obsahuje také funkce, které jsou spjaty s analýzou dat získaných z protokolu. Do protokolu se ukládají veškeré aktivity a chování studentů v systému, obzvláště při výuce a testování. Jedná se o dobu strávenou u jednotlivých částí výuky, posloupnost zvolených jednotek výuky či dočasné i konečné odpovědi studentů během testování včetně vyhodnocení. Vyhodnocováním těchto dat jsou získávány různé číselné statistiky, znalosti nebo grafové vyjádření průchodu výukou. Tyto výsledky jsou analyzovány a výsledné úpravy v generování výuky poté pozitivně ovlivňují její kvalitu.

Obecně je LMS definován jako softwarová aplikace založená na webové technologii, používaná k přípravě, realizaci a vyhodnocení zpětné vazby konkrétního procesu učení. LMS poskytuje učitelům metody, jak vytvořit a zobrazovat obsah vzdělávacích materiálů, monitorovat účast studentů a hodnotit jejich výsledky. LMS může také poskytnout studentům možnost využít interaktivní prvky, jako jsou strukturované diskuse, videokonference a diskusní fóra. [6] Ovšem žádný LMS zatím neřešil individuální výuku na míru konkrétnímu studentovi [1].

Výhody LMS [1]:
- Systém prezentuje informace mnoha způsoby, integruje práci s textem, obrázky, zvukem, videem apod. Umí to vše řídit, evidovat, vést statistiky a vyhodnocovat.
- Elektronické učebnice jsou oproti tištěným verzím průběžně modifikovatelné.
- Učebnice mohou být doplněny nejrůznějšími typy multimédií.
- Studenti mohou využívat LMS online kdekoliv a kdykoliv.
- Automatické vyhodnocování odpovědí umožňuje aktuální okamžitou zpětnou vazbu a evidenci o průběhu studia.
- Analytické metody umožňují dlouhodobou zpětnou vazbu statistickým vyhodnocováním výukového procesu.

Nevýhody LMS [7]:
- Primární orientace pro využití LMS je spíše na středních a vysokých školách či u vzdělávání dospělých.
- Riziko, že dojde k úplné eliminaci role učitele.
- Existuje mylné povědomí o možnosti plného nahrazení "klasické"výuky e-learningovými nástroji.

2.4 Teorie adaptivní výuky

Samotná výuka a proces výuky jsou chápany jako posloupnost kroků učení. Tyto kroky by měly být přítomny v každém výukovém celku (kurzu, lekci či hodině). Jsou základem pro tvorbu teoretického modelu adaptivní výuky. Teorie adaptivní výuky vychází ze dvou takových výukových procesů obsahujících posloupnost událostí [1]. První proces je podle psychologa R. Gagného rozdělen na 9 částí [8]:

1. Získání pozornosti – představení řešeného problému či situace, ukázání chybného postupu
2. Informování žáků o cílech – popis cíle lekce a čeho studenti dosáhnou, umožnění studentům přichystat se k naslouchání
3. Vyvolání předchozího naučení – připomenutí dosavadních znalostí žáků, vytváření vztahů mezi nabytými znalostmi a znalostmi z aktuální lekce
4. Prezentace materiálu – rozdělení výuky na malé části, prezentace materiálu s uplatněním výukových strategií
5. Provázení učením – formulace pokynů a ukázaní způsobů, jak se danou látku naučit
6. Iniciativa a povzbuzení k výkonu – procvičování studentů, aby vykonávali úlohy samostatně s využitím nových znalostí
7. Poskytnutí zpětné vazby – konkrétní analýza a reakce na studentovy odpovědi, vysvětlení správné odpovědi
8. Ohodnocení výkonu – otestování studentů pro ověření úspěšnosti výuky
9. Zlepšení uchování v paměti – zopakování lekce, informování studentů o podobných situacích či možnostech dalšího procvičení

Druhý proces, ze kterého adaptivní výuka vychází, je Bloomova taxonomie vytvářející hierarchie studentem vykonávaných činností, jež jsou postupně stupňovány podle obtížnosti, obsáhlosti či náročnosti. Proces je rozdělen na tyto části [9]:

1. Zapamatování si – vybavování si relevantních znalostí z dlouhodobé paměti
2. Porozumění – interpretování získané informace vlastními slovy
3. Aplikování či použití znalostí – použití naučeného postupu v dané situaci
4. Analýza informace – rozebrání celku na složky a určení, jaký mají účel a které k sobě patří
5. Syntéza – schopnost provést hodnocení na základě daných kritérií
6. Hodnocení – opětovná organizace částí nabytých znalostí, aby vytvořily nový funkční celek

Dále jsou základním kamenem tvorby pravidel a adaptivních algoritmu pro přiřazení materiálů konkrétním studentům zásady J.A.Komenského a Kolbův cyklus [1]. Mezi nejdůležitější zásady Komenského se řadí názornost (zapojení co největšího počtu smyslů do učení), systematičnost a soustavnost (logické uspořádání učiva). Dále postupný výklad (od známých vědomostí k neznámým – zásada předchozích znalostí), přiměřenost (stanovené cíle jsou přizpůsobené věku, znalostem a vyspělosti studentů) a trvalost (pravidelné opakování a procvičování) [10].

Kolbův cyklus je rozdělen na dvě hlavní činnosti, v první probíhá samotná aktivita studenta či řešení úkolu a v druhé části reflexe (neboli rekapitulace či ohlédnutí za předchozí aktivitou), přičemž reflexe má tři fáze. V tomto cyklu je kladen důraz na rozvíjení dovedností. Kolb také rozlišuje čtyři styly učení a těmi lze v adaptivním e-learningu vyjádřit, který styl učení daný student upřednostňuje [1]:

1. Divergující (více variant řešení problému) – zvukové nahrávky, prožitky, případové studie, spojení s praxí
2. Asimilující (vstřebává různé údaje) – tvorba modelů, vytváření vzorových řešení
3. Konvergující (schopen najít jedno řešení) – logické zamyšlení a praktická zkušenost, aplikace nových myšlenek do praxe
4. Akomodující (adaptace na nové situace, intuice) – propojení na praxi, úkoly z praktického života

2.4.1 Individualizace výuky

V případě LMS Barborka 4 je upřednostněno zaměření na individualizaci výuky vztaženou k osobnosti studenta a přizpůsobenou každému studentovi podle jeho aktuálních možností a schopností. Tato forma výuky, kdy učitel vyučuje právě jednoho žáka je velmi efektivní. Virtuální učitel má rozsáhlý přehled o individuálních potřebách studenta, upřednostňuje jeho samostatnou a dobrovolnou práci. Virtuální učitel musí také držet řád při výuce. Takový výukový styl se dá nazvat jako demokratický až místy autokratický.

Avšak kvůli individualizaci výuky je potřeba v expertní části implementovat metody, které jsou realizovatelné v prostředí e-learningu. To jsou metody závislé na některých vlastnostech. Studenti jsou velmi odlišní, at' už z hlediska zdravotního stavu, handicapu, přístupu k učení, intelektuálních schopností či vlastních dříve nabytých zkušeností atd.

V prvé řadě systém musí zjistit výchozí parametry a údaje testováním studenta pomocí dotazníků. Na základě zvoleného učebního stylu systém pomocí adaptivních algoritmů vybírá vyučovací styl a přizpůsobuje obsah učebních materiálů. Míchá a vybírá nejvhodnější části adaptivních materiálů. Pokud však prvotní učební styl studenta není přesný, jelikož jsou často jeho odpovědi nesprávné, a v průběhu výuky se zdá, že je zvolen dle vstupního dotazníku nepřesně, systém vhodně mění jeho aktuální učební styl. Jen díky přesným informacím o jeho charakteristikách a okamžitých znalostech dochází k efektivnímu procesu výuky.

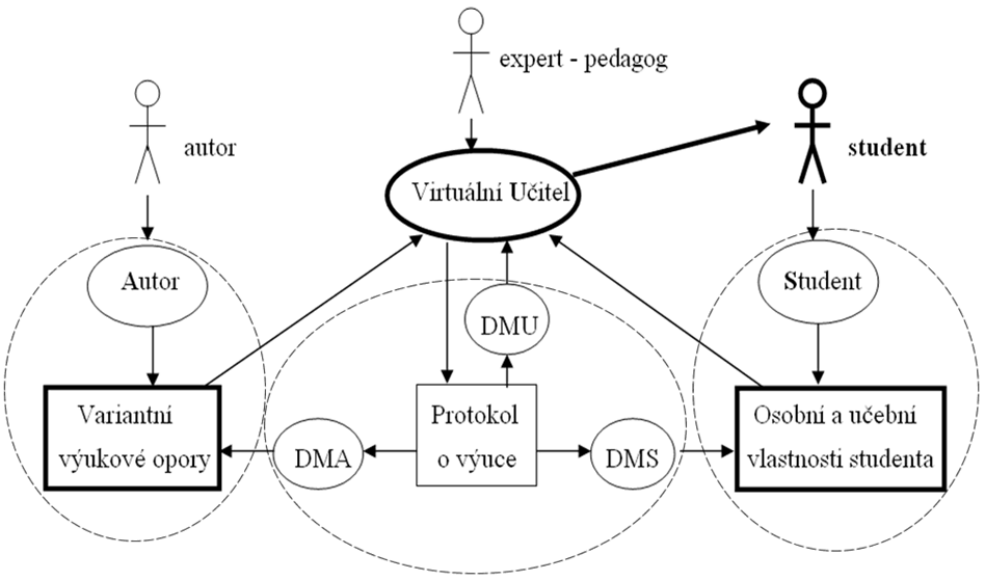
Důvody zavedení individualizace výuky:
Důvody pro individualizaci jsou na straně studentů i učitelů. Z hlediska studentů již existuje řada typů výuky, které jsou zaměřeny na individuální studium. Jedná se například o formy celoživotních kurzů, distančního či kombinovaného studia atd. Dalším důležitým důvodem je, že studenti vysokých škol již vlastně preferují samostudium sami. Prezenční formy se jich účastní čím dál méně, využívají tak nepovinné výuky, možnosti počítačů a dostupnosti výukových materiálů na internetu. Studenti mnohdy vyhledávají počítače, jelikož se s nimi setkávají prakticky každý den. Kdykoliv se potřebují něco dozvědět nebo vyhledat, použijí internet. Dobré výukové programy mohou být pro ně dokonce i hrou,

neunavují, podporují jejich fantazii a dokážou je dobře motivovat. V hromadné výuce se mohou někteří studenti nudit anebo nemusejí rozumět probírané látce. V obou těchto případech je pro ně výuka demotivující. Naopak personalizovaná e-learningová adaptivní výuka navazuje na aktuální dovednosti studenta, dokáže vypustit látku, kterou student dobře ovládá, nebo se zaměřuje a podrobně vysvětluje to, co student přesně nechápe.

Z hlediska učitelů také existuje několik důvodů, proč by se měla zavést individualizace e-learningové výuky. Kvůli velkému počtu studentů ve výuce se učitel nedokáže jednotlivě věnovat méně schopným studentům, nebo naopak těm, kteří jsou více motivovaní a rádi by se dozvěděli více. Pomocí kombinace klasické a e-learningové výuky se mohou snížit počty přednášek a tím vycházet studentům vstříc. Nakonec je také výuka desítek i stovek studentů časově náročná, což si přirozeně vynucuje automatizaci rutinních prací učitele včetně zkoušení a testování [11].

2.4.2 Model adaptivní výuky

Základní myšlenka teorie adaptivní výuky byla převzata z tzv. programového učení [1]. V principu jde o rozdělení učiva do velmi malých částí, kde každá obsahuje jednu základní informaci, přičemž se v tomto případě vyskytují ve více variantách. Tyto úseky výuky konstruuje a navrhuje přímo učitel – autor, a tím vytváří výukový a testovací obsah. Dále jde o ověřování těchto částí a o následnou reakci na studentovo pochopení látky. V tomto případě virtuální učitel průběžně kontroluje pochopení informací, vhodně reaguje nejen na správné a chybné odpovědi studenta, ale také na jeho obecné osobní charakteristiky.



Obrázek 1: Model výuky

Z praktického hlediska je teoretický model výukového systému rozložen na tři subsystémy (studentský, autorský a řídící) [1, 5]. Každý subsystém pracuje s vlastní částí

databáze, přičemž existuje mnoho vazeb mezi databázovými entitami. Takové rozložení struktury systému lze vyjádřit schématem na obrázku 1.

Subsystém Student
Jelikož je ústřední osoba celého systému student, byl pro něho vytvořen samostatný subsystém. Ten podle teorie obstarává diagnostiku aktuálních znalostí a učebních charakteristik studenta, průběžnou výuku a testování studenta.

Subsystém Autor
Dále je třeba strukturalizovat, ukládat a modifikovat výukové opory a jejich metadata, která popisují části výuky, zdali se jedná o definici, motivaci či samostatný úkol apod. Proto byl zvlášť vytvořen subsystém pro autora adaptivních výukových opor, jenž obsahuje funkce pro tvoru těchto opor.

Subsystém Expert (virtuální učitel)
Nakonec je potřeba podle návrhu definovat adaptivní algoritmy pro formulování optimálního personalizovaného studijního prostředí a zaznamenávat průběh výuky. To probíhá v subsystému virtuálního učitele. Pro generování výuky na míru využívá všechny informace o výukové struktuře a o studentovi. Je zde obsažen expertní systém, jenž je založen na pedagogicko-psychologických znalostech. Expert sestavuje plán optimálního průchodu výukou a pro sestavení optimálního výukového stylu používá elementární pedagogická pravidla.

Student si sám podle sebe může zvolit průběh výuky v jiném pořadí, než je mu nabízeno. Dlouhodobým sledováním jeho akcí (včetně časových úseků strávených nad výukou, řešením úloh atd.) ukládaných do protokolu a následnou data-miningovou analýzou činnosti studenta z protokolu systém získává zpětnou vazbu. Tímto zdokonaluje pravidla pro vytváření výuky.

2.4.3 Učební styly a charakteristiky studenta

Existuje mnoho různých definic a klasifikací učebních stylů studenta, ty jsou ale nejednotné. Vesměs se v definici stylu učení hovoří o typických způsobech jednání, predispozicích či vlastnostech, které se týkají situací vyučování či učení, a také, že se jedná o proces zpracovávání informací ovlivňující charakteristiky osobnosti [1].

Zkoumáním různých vlastností studentů se postupem času zabývalo mnoho autorů. Výsledkem těchto zkoumání bylo rozdělení učebních stylů podle mnoha kritérií. Všechny vlastnosti jsou na sobě navzájem nezávislé, a tedy hodnoty mohou existovat v jakékoliv kombinaci a tvořit tak charakteristický učební styl [1]. Vlastnosti jsou také buď jen vrozené anebo získané výchovou. Kombinace vlastností u každého studenta není stálá, ale dynamicky se mění s přirozeným vývojem studenta [1]. V případě teorie adaptivního e-learningu byly vybrány některé vlastnosti studenta a učební styl je dán kombinací hodnot těchto vybraných vlastností.

Vlastnosti použitelné v rámci e-learningu jsou rozděleny do následujících pěti skupin [1]. První skupina pojmenovaná „smyslové vnímání"popisuje, jaké smysly student nejvíce používá, a obsahuje informaci o formě podání učiva, která studentovi

nejvíce vyhovuje. Druhá skupina „sociální aspekty"se zabývá zapojením studenta do společnosti při učení. Následující skupina s názvem „afektivní aspekty"se zaměřuje na motivaci, city a postoje studena ovlivňující průběh učení. Další skupiny popisující „taktiky učení"jsou nejrozsáhlejší a obsahují čtyři detailnější charakteristiky. Jedná se o „systematičnost učení", která popisuje potřebu použití kroků učení podle manuálu, o „způsob učení"rozlišující preferenci teorie či praktického experimentování, o „postup učení"popisující, s jak velkou informací může student pracovat, a o „pojetí učení"charakterizující, jakou hloubku učební látky má potřebu studovat. Poslední vlastnost se nazývá „autoregulace"a ta určuje míru samostatnosti. To znamená, jak moc je student schopen sám řídit své učení. Shrnutí všech skupin s použitými vlastnostmi je zobrazeno v tabulce 2.

Název vlastnosti		**Hodnoty**
Smyslové vnímaní:		verbální – vizuální – auditivní – kinestetické
Sociální aspekty:		rád pracuje sám – ve dvojici – ve skupině
Afektivní aspekty:		motivace ke studiu vnitřní – vnější
Taktiky učení:	*Systematičnost s póly:*	upřednostňuje řád – volnost
	Způsob s možnostmi:	teoretické odvozování – experimentování
	Postup s možnostmi:	analytický – holistický
	Pojetí s možnostmi:	hloubkový – strategický – povrchový
Autoregulace s póly:		pracuje dle pokynů – samostatně

Tabulka 2: Vlastnosti studenta

2.4.3.1 Popis vlastností

Dále jsou zvlášť popsány jednotlivé vlastnosti výše uvedené společně s kvantifikací [1, 5, 12]. Pro numerické ohodnocení hodnot typů vlastností se používají stupnice celých čísel v rozmezí <0,100> nebo <-100,100> pro opačné póly. Každého studenta charakterizuje jedna hodnota z každého typu vlastnosti.

Smyslové vnímaní

Lidé mají typicky danou kombinaci smyslů, nazývající se multimodální vnímání, a pouze málo osob je jednoho typu. Ovšem vychází se z toho, že jeden smysl převládá. Smyslové vnímání tedy tvoří vektor obsahující hodnoty 4 složek smyslového vnímání. Tyto hodnoty se pohybují v intervalu <0,100> a jejich součet vždy dává hodnotu 100.

Verbální – Je zvýhodňována textová prezentace informace pomocí slov. Student preferuje čtení a psaní.

Vizuální – Jedná se o preferenci grafického ztvárnění objektů místo slovního sdělení. Student si pamatuje a představuje to, co vidí.

Auditivní – Tento typ studena používá hlavně sluch, vnímá poslechem a pamatuje si to, co slyšel nebo říkal.

Kinestetické – Jedná se o pohybový či motorický typ vnímání. Student si v tomto případě umí snadno zapamatovat věci, se kterými mohl manipulovat či si je osahat. Snáze se učí poznatky, které by mohl raději využít v praxi nebo v realitě.

Sociální aspekty
Pro sociální aspekty je použita stupnice <0,100>.

Rád pracuje sám – Student preferuje samostatné učení bez kontaktu s okolím, zaměřuje se na své myšlenky a pocity. Tato definovaná vlastnost má hodnotu 0.

Ve dvojici – Potřebuje mít někoho po ruce, s kým může diskutovat o daném tématu, nebo někoho, kdo mu může pomoct či poradit. Tato vlastnost má hodnotu 30.

Ve skupině – Potřebuje mít při učení kontakt s lidmi, učí se ve skupinách a vyhledává spolupráci s ostatními. Tato vlastnost má maximální hodnotu 100.

Afektivní aspekty
V tomto případě se jedná o vlastnost obsahující póly (má ke studiu odpor vs. má velký zájem), a tak je použita stupnice s intervalem <-100,100>.

Nemotivovaný – Student může být nemotivovaný buď vlastním nezájmem anebo nepochopením učiva. Nezájem je způsobem tím, že nevidí vlastní uplatnění v praxi nebo ho předmět nebaví. Naopak nepochopení učiva může být způsobeno špatným učením a neschopností. Obecně může cítit odpor k učení, což je ohodnocenou minimem -100 nebo může projevovat jen nezájem a tento stav má hodnou -50.

Neutrální – Student je lhostejný a je mu vyloženě jedno, co se učí a jak. Hlavním cílem je úspěšně ukončit předmět s minimem naučených znalostí. Vlastnost má hodnotu 0.

Motivovaný – Má zájem o studium, vidí perspektivy a uplatnění v tom co studuje. Kvantifikace menší motivace je ohodnocena 50, kdežto velký zájem o studium zastupuje maximální hodnota 100.

Taktiky učení:
Hodnoty systematičnosti, způsobu a postupu zpracování informací se udávají v intervalu <0,100>. Hodnota pro vlastnost pojetí zpracování je udávána v intervalu <-100,100>.

Systematičnost

Potřeba volnosti – Student se učí po svém s maximální volností, učební materiály si náhodně prochází sám a vybírá si své způsoby řešení úkolů, taktéž si sám hodnotí své výsledky. Je to minimální hodnota systematičnosti, tedy 0.

Potřeba řádu – Postupně se učí podle přesných pokynů a podle návodu určujícího jednotlivé kroky. Než však začne plnit úkol, je mu nejdříve poskytnuto podrobného vysvětlení problému. Tato definovaná vlastnost má hodnotu 100.

Způsob zpracovávání informací
V tomto případě jsou obě hodnoty vyjádřeny jako dvě nevylučující se samostatné vlastnosti. Proto způsob, jakým může student zpracovat výuku, je vyjádřen vektorem se dvěma složkami.

Teoretik – Při učení se zastavuje, nejprve přemýšlí o souvislostech s danou látkou a odvozuje jiné poznatky. Dělá krátká shrnutí nabytých vědomostí.

Experimentátor – Student pochopí význam až pomocí aktivního experimentu nebo praktické aplikace na jinou věc. Také zkoumá reálnou užitečnost nabyté informace.

Postup zpracovávání informacíí
Zde se taktéž obě dvě vlastnosti nevylučují. Student může ovládat různou mírou oba postupy. Postup zpracování informací je tedy vyjádřen formou vektoru se dvěma složkami.

Detailistický – Student zpracovává informace z učebních materiálů po malých částech, zaměřuje se na detaily a má často problém pojmout všechny části jako jeden celek.

Holistický – Na rozdíl od detailistického postupu při učení zachycuje poznatky ve větším celku a hůře rozlišuje detaily.

Pojetí zpracovávání informací

Patologicky povrchové – Student se učí danou látku nazpaměť bez snahy porozumět, o co se vůbec jedná. Této definované vlastnosti byla přiřazena hodnota -100.

Povrchové – V tomto případě jen vyhoví základním povinným klasifikačním požadavkům a přitom nemá zájem o studium. Jde mu v podstatě jen o správný přednes látky, aniž by chápal obsah. Hodnota je -50.

Strategické – Snaží se pouze rychle dosáhnout co nejlepšího výsledku za vynaložení co nejmenšího úsilí a vytváří způsob, jak na to. Chytře pozoruje učitele při zadávání podmínek a hodnocení. Toto pojetí má hodnotu v intervalu 0-50.

Hloubkové – Při hloubkovém pojetí se student snaží velmi aktivně porozumět látce a sám má potřebu důkladně a detailně pochopit význam nových poznatků a souvislost s těmi, které již zná. Hloubkové pojetí má nejvyšší hodnotu 100.

Autoregulace
Vlastnost autoregulace nabývá hodnot z intervalu <-100,100>.

Nepřizpůsobivý – Student neplní úlohy a nemá zájem se přizpůsobovat jakýmkoliv podmínkám učitele. V tomto případě byla určena nejmenší opačná hodnota -100.

Direktivní řízení – Základem je nesamostatnost studenta a tedy závislost na učiteli. Ten se snaží potlačit jeho odpor. Výuka probíhá stylem výkladu a přesvědčování o správnosti či zvýraznění jeho nedostatků. Tento typ autoregulace má hodnotu -50.

Sdílené řízení – Student je zainteresovaný ve výkladu učitele. Ten ho provádí výukou a se zájmem diskutují. Vlastnosti sdílené řízení je přiřazena hodnota 0.

Volné vnější řízení – Plně se zabývá sebou, podílí na rozvoji vlastních vědomostí s možnosti přispění učitele, jenž je vůči němu v roli partnera. Výuka je volná a jeho výsledky jsou hodnoceny podle praxe. Tato vlastnost má hodnotu 50.

Řízení sama sebe – V tomto případě se student učí a řeší úkoly zcela samostatně, vytyčuje si své postupy i cíle. Poznatky získává z praxe a jeho učitel je jen konzultant. Autoregulace je zde maximální a nabývá hodnoty 100.

2.4.3.2 Virtuální student

Jelikož individuálních kombinací všech možných hodnot u 14 používaných vlastností je opravdu velmi mnoho, dokonce při zobecnění na 2 hodnoty pro jednu vlastnost se jedná o 214 učebních stylů, je nereálné vytvářet pro každou unikátní kombinaci charakteristik individuální výukový proces. Proto bylo v teorii adaptivní výuky zavedeno zobecnění na rozumný počet virtuálních studentů. Jedná se o několik typických nadefinovaných virtuálních studentů podle pedagogických znalostí. Virtuální student tedy obsahuje jednu kombinaci vlastností typickou pro množinu studentů s podobnými charakteristikami. Virtuální učitel generuje výuku na míru studentům právě podle jejich přidělených virtuálních studentů.

Podle studentem vyplněného dotazníku se musí určit jeho aktuální virtuální student, se kterým má nejpodobnější vlastnosti. Změny učebního stylu studenta jsou však povoleny, takže se v případě změny vlastností mění i virtuální student. Pokud však virtuální student nevyhovuje, jelikož přiřazené množině studentů nezprostředkovává optimální individuální výukový proces, systém po analýze protokolu může změnit přímo charakteristiku daného virtuálního studenta.

2.4.4 Výukové opory

Výuková opora je vždy vztahována k předmětu a je obecně složena z hierarchické struktury. Systém také podporuje i neadaptivní formu, ale je zaměřen hlavně na formu adaptivní. Obsahem opory předmětu jsou jednotlivé elementární části výukového procesu. Výukový proces je rozdělen podle R. Gagného na 9, již zmíněných, elementárních událostí (začátek kapitoly 2.4). Tyto události by měly být přítomny v každém výukovém celku (kurzu, lekci či hodině). Následující tabulka 3 ukazuje aplikaci zásad tohoto procesu do výukových materiálů [1].

Událost (dle Gagného teorie)	*Součásti elementární části ve výukové opoře*
Získání pozornosti	motivační část
Informování žáků o cílech	formulace cílů
Vyvolání předchozího naučení	vstupní testování znalostí, případně prerekvizity rámce
Prezentace materiálu	teoretická část
Provázení učením	sémantická, vysvětlovací část
Iniciativa a povzbuzení k výkonu	příklady z praxe
Poskytnutí zpětné vazby	autotestování – otázky a příklady
Ohodnocení výkonu	výsledek testů, klíč k neřešeným příkladům
Zlepšení uchování v paměti	fixační, opakovací, procvičovací část

Tabulka 3: Aplikace zásad do výukových materiálů

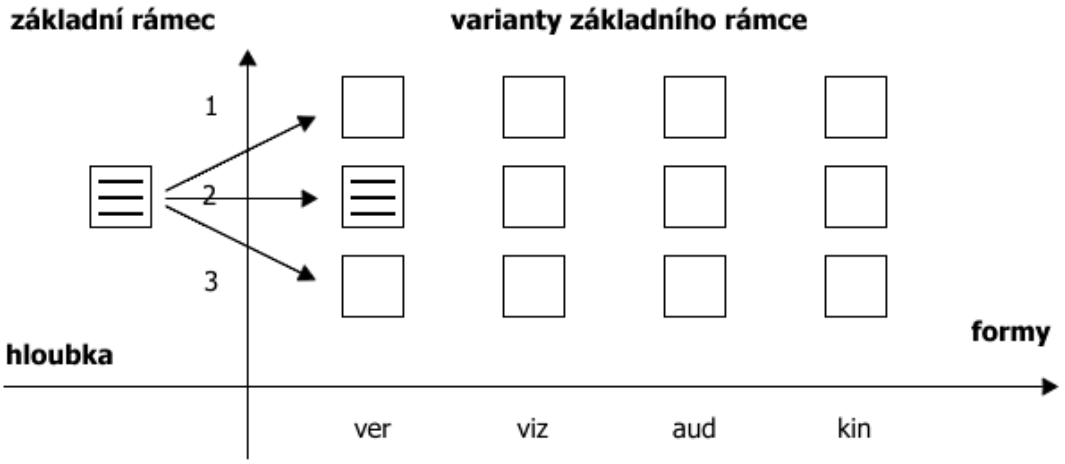
Při formulaci cílů v jednotlivých částech výkladu je postupováno v souladu se zmíněnou Bloomovou taxonomií. Autoři opor při vytváření úloh vycházejí z doporučení a taxono-

mie učebních úloh dle Tollingové. Při tvorbě opor jsou zohledněny i další zmíněné zásady Komenského [1].

2.4.4.1 Struktura

Běžná skripta obsahuji statické informace a jsou stejná pro všechny studenty. Pro účely adaptivní výuky jsou skripta rozdělena na dílčí části, nesoucí elementární informace uvedené v několika formách, z důvodu, aby systém mohl podávat látku různými způsoby či v různých smyslových variantách. Skripta jsou strukturována do kapitol, podkapitol a odstavců, které jsou však různé délky. Proto byla za jednotku výuky zvolena lekce. Délka obsahu jedné lekce představuje délku jedné běžné vyučovací hodiny a ta může být shodná s kapitolou. Dokonce i učitelé prezenční formy studia si dělí vyučovací látku podobným způsobem. Lekce je složena z elementárních částí nazvány jako rámce.

Rámce se od sebe odlišují převažujícím typem smyslového vnímaní (verbální, vizuální, auditivní a kinestetický typ) a hloubkou pojetí (hloubkové, strategické a povrchní). Každý rámec existuje v jedné kombinaci typu smyslového vnímání a hloubky pojetí. Jedné této kombinaci se říká varianta. Rámec může logicky mít aktuálně jednu z celkových 12 variant (obrázek 2). Pro autora opory je ovšem obsah pro všechny varianty stejný, jen je podán odlišnou formou. Úkolem autora je tedy pro danou elementární informaci vytvořit 12 variant a vyrobit tak všechny druhy výkladu. Pokud to však není potřebné nebo vhodné, není vždy nutné vytvořit všechny varianty [13].



Obrázek 2: 12tice variant

Hloubka s číslem 1 je určena pro výborné chápavé studenty a obsahuje rozšiřující informace, hloubka s číslem 2 je výchozí s nejčastěji používaným výkladem a pod hloubkou 3 se nachází podrobný výklad.

Varianta však není konečný jednolitý celek, ale rozděluje se na další jednotlivé části z hlediska fáze vyučovacího procesu, jelikož varianty nedokážou pokrýt všechny potřebné rozdílnosti ve vyučovacím stylu [5]. Těmto nejmenším částem se říká vrstvy. Existuje několik typů vrstev, přičemž v jedné variantě se může vyskytovat maximálně jedna vrstva daného typu. V případě, že se v aktuální variantě rámce nachází více vrstev

stejného typu, jedná se o tzv. multivrstvu. Všechny definované typy vrstev jsou popsány v kapitole 2.4.4.2.

Zatím zde byla popsána jen struktura dat, reálná data se nacházejí v komponentách. Jedna vrstva je složena z takových datových komponent, které přímo obsahují zobrazovaná data. Existuje také několik typů komponent (text, zvuk, video, textový soubor, PDF soubor, obrázek, animace čí applet).

2.4.4.2 Typy vrstev

V této části je uveden popis jednotlivých typů vrstev rozdělených do 3 skupin [14].

Výkladové

Tato skupina obsahuje vrstvy s vlastním výkladem probírané látky.

T (teoretická) – obsahující teorii: definice, pojmy, pravidla atd.
S (sémantická) – vysvětlující zaváděné pojmy, formálně popsanou teorii, doplňující informace k teoretické vrstvě a souvislosti plynoucí z teorie
F (fixační) – usnadnění zapamatování teorie pomocí opakování, použitím jiných formulací či zasazením do širšího kontextu
R (řešené příklady) – řešené „školní " příklady na využití teorie; vzor k řešení úkolů
P (praktická) – řešené příklady z praxe využívající teoretické znalosti

Testovací

Skupina vrstev testovacího typu zahrnuje vrstvy pro průběžné testování znalostí a jejich osvojení. Každá otázka obsahuje zadání i jednoznačné odpovědi.

O (otázky) – teoretické otázky z probrané látky
U (úlohy) – školní úlohy k řešení
X (praktické úlohy) – úkoly z praxe

Ostatní

C (cíle) – formulované cíle lekce nebo rámce
M (motivace) – motivující informace, zdůvodňující přínos studia
N (navigace) – pedagogické a organizační informace; průvodce látkou; doporučený postup při studiu
L (literatura) – doporučená literatura
Z (neadaptivní) – určení, že se jedná o vrstvu pro neadaptivní verzi

2.4.4.3 Testovací otázky a odpovědi

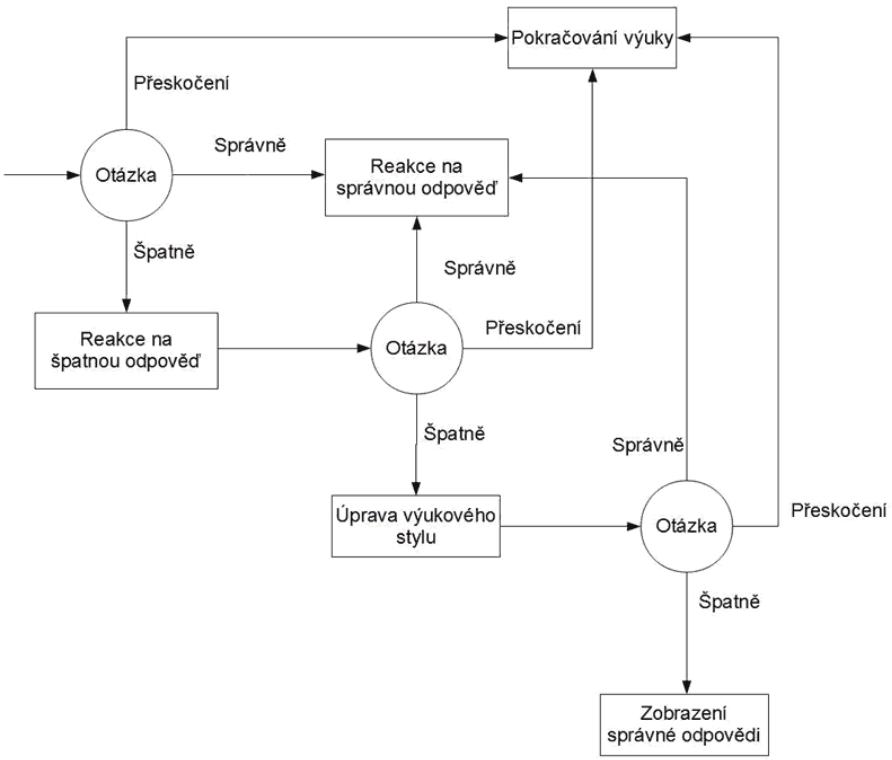
Jeden vyučovací proces rámce může ve variantě obsahovat více průběžných testovacích otázek. Je to dokonce autorům doporučeno, a proto jsou otázky vesměs definovány jako multivrstvy [14]. Stejné testovací otázky mohou být použity nejen při výuce, ale i při autotestování a ostrém testování. Pro tyto účely jsou otázky obecně popsány různými metadaty, těmi jsou například typ vrstvy, pořadí vrstvy v rámci výkladu, pořadí vrstvy v multivrstvě, typ vyhodnocení, body aj.

Ke každé otázce také může existovat libovolný počet odpovědí. Očekávané odpovědi a příslušné reakce jsou v systému uvedeny v úplném a přesném znění. Evidují

se 3 typy otázek a odpovědí – variantní, tvořené a speciální. U variantních otázek se jedná pouze o určení správnosti (zaškrtnutí) zobrazených odpovědí pod otázkou. Variantních odpovědí může být libovolný počet, jelikož se ke každé otázce odpovědi vybírají. U tvořené otázky musí student sám vyplnit svou slovní odpověď. U tohoto typu může rovněž existovat libovolný počet odpovědí, jelikož na otázku může existovat více správných odpovědí či typických polosprávných a nesprávných odpovědí, pro které je vhodné uvádět reakce. Systém porovnává tvořené odpovědi studenta s definovanými očekávanými a hledá příslušnou shodu. U tvořených otázek existuje ještě jeden zvláštní typ odpovědi, a tím je doplňovačka (doplňování částí textů či výběr textu ze seznamu) vhodný pro jazykové kurzy. Doplňovačka je jeden z typů tvořených odpovědí, jelikož se z vyplněných slov stává vektor slov. Speciální otázky vyžadují speciální softwarové nástroje pro formulaci i vyhodnocení odpovědi. U odpovědí se taktéž ukládají metadata popisující například typ odpovědí, pořadí odpovědi, reakci k odpovědi, definovaný text aj. Všechny dosud definované typy odpovědí popisuje následující tabulka 4 [14].

Typ odpovědi	**Zkratka**	**Vysvětlení**
Variantní		
	V1	variantní s 1 správnou odpovědí – bez manipulace s pořadím
	V1m	variantní s 1 správnou odpovědí – s mícháním pořadí
	Vn	variantní s více správnými odpověďmi – bez manipulace s pořadím
	Vnm	variantní s více správnými odpověďmi – s mícháním pořadí
Tvořené		
	Tc1	číslo
	Tcn	množina čísel oddělená čárkou, nezáleží na pořadí
	Tcv	vektor čísel oddělených čárkou, záleží na pořadí
	Ts1	slovo = řetěz znaků bez čárky a mezery, diakritika se ruší
	Tsm	množina slov oddělených čárkou, diakritika se ruší
	Tsv	vektor slov oddělených čárkou, diakritika se ruší
	Td1	slovo s diakritikou = řetěz znaků bez čárky a mezery
	Tdm	množina slov s diakritikou oddělených čárkou
	Tdv	vektor slov s diakritikou oddělených čárkou
Speciální		
	Sou	soubor, určený pro vyhodnocení „ruční “
	Sav	algebraický výraz zapsaný sekvenčně dle programovacího jazyka
	Scj	číslo s měrnou jednotkou odděleny mezerou, jednotka jako algeb. výraz
	Suh	graf typu uzly – hrany

Tabulka 4: Typy odpovědí u testovacích otázek



Obrázek 3: Reakce virtuálního učitele

2.4.5 Virtuální učitel a výukový styl

Virtuální učitel zprostředkovává hlavní řízení, generování výuky a provázení studenta výukou. Obecně je vyučovací styl učitele definován jako jeho svébytný postup, jímž vyučuje, soubor činností, které učitel jako jedinec uplatňuje ve vyučování [15]. Vyučovací styl se promítá do učitelových vyučovacích strategií, do volby výukových metod a do učitelova pojetí výuky [1]. V případě tohoto systému je učitelem virtuální učitel, jehož výukový styl je adaptivní. Řídí výuku a určuje optimální postup zobrazení jednotlivých vrstev a variant. Nejprve musí podle učebního stylu (US) přiděleného virtuálního studenta určit osobní výukový styl (OVS), pro který ale nemusí vždy existovat varianty rámců, a tak ho ještě dále aplikuje na aktuální výukový styl (AVS) lekce použitý pro vygenerování struktury výukových materiálů.

Speciální expertní algoritmus tedy určuje z US pomocí definovaných elementárních pravidel optimální OVS studenta, což je úplná posloupnost vrstev s nastavenou hloubkou [1]. Další algoritmus poté porovnává strukturu každého rámce aktuální lekce s OVS. Pokud neexistuje daný typ vrstvy, nahradí se jinou dostupnou vrstvou v nejbližší hloubce a formě. Vrstvu může také vynechat. Výsledkem algoritmu pro AVS je formálně stejná posloupnost jako u OVS, ale je rozčleněná opakovaně speciálně pro každý aktuální rámec lekce [1].

Elementární pravidla slouží pro výběr nejvhodnější smyslové varianty a k určení aktuálního pořadí typů vrstev a hloubky pro každý teoreticky úplný rámec. Tato definovaná pravidla vycházejí ze statických i dynamických vlastností studenta. Do systému je vkládá odborník (učitel – expert) [1].

Také při reakci na odeslanou správnou odpověď studenta u testovací otázky ve výuce se postupuje dle AVS. V opačném případě, pokud odpoví chybně, virtuální učitel aktivuje nový proces řízení, který vybírá reakci, vhodně řeší změnu některých vrstev, zobrazí správnou odpověď nebo reaguje jiným způsobem. První chybu u variantních otázek systém studentovi dovoluje opravit bez reakce. Při druhé chybě systém zobrazí reakci na špatnou odpověď, při třetí chybě učitel provede úpravu výukového stylu s vysvětlením otázky a při čtvrté chybě vypíše studentovi přímo správnou odpověď s vysvětlením [1]. Tento proces je uveden na obrázku 3.

3 Projekt Barborka 4

Barborka verze 4 je programový systém typu webové aplikace pro adaptivní řízení individualizované výuky a její správu. Tím se řadí do skupiny systémů LMS. Mimo tyto hlavní funkcionality ještě obsahuje autorské ukládání výukových opor, testování studentů, správu předmětů, správu úkolů, správu testů, evidenci studentů, jejich aktivit a výsledků. Obsahuje také ostatní funkcionality spojené s individualizovanou výukou a podporou řízení výuky. Barborka 4 je součástí dvou projektů spolufinancovaných z prostředků Evropské unie ESF a státního rozpočtu České republiky. Tyto projekty patří pod operační program „Vzdělávání pro konkurenceschopnost“, byly řešeny a implementovány na vysokých školách VŠB-TUO FEI na katedře informatiky a OSU PdF na katedře informačních a komunikačních technologií. Implementační část softwarového systému byla intenzivně vykonávaná od září 2011 a schůze programátorů s metodiky probíhaly rok a půl v kuse prakticky každý týden. V současnosti je LMS Barborka 4 provozován jako veřejně dostupný systém na internetu běžící na serveru katedry informačních a komunikačních technologií Ostravské univerzity[1].

Řešitelské role:
Na tomto projektu se podílí mnoho odborníků v pěti řešitelských rolích.
- *Metodici* (pedagogové, psychologové, informatici) jsou autory různých analýz, teorie adaptivní výuky a odborníky na procesy uvnitř systému s teorií spojené. Uplatňují tyto metody při návrhu a implementaci, ověřují správnost a provádí odborné konzultace.
- *Autoři* opor mají na starost návrh a tvorbu pilotních výukových adaptivních opor.
- *IT technici* vytvářejí multimediální soubory, jež se používají v adaptivní výuce.
- *Programátoři* vyvíjejí samotný adaptivní LMS.
- *Analytici data-miningových metod* provádí analýzy a dolování dat z protokolu.

Hlavičky obou projektů:
Operační program Vzdělávání pro konkurenceschopnost
Název: Personalizace výuky prostřednictvím e-learningu
Číslo: CZ.1.07/2.2.00/07.0339
Realizace: Vysoká škola báňská - Technická univerzita Ostrava
[16]

Operační program Vzdělávání pro konkurenceschopnost
Název: Adaptivní individualizovaná výuka v e-learningu
Číslo: CZ.1.07/2.3.00/09.0019
Realizace: Ostravská univerzita v Ostravě
[17]

[1] Aktuální verze je dostupná na webu http://kik.osu.cz/barborka4

3.1 Historie LMS Barborka

Historie systému, jehož název vznikl podle patronky všech horníků svaté Barbory, sahá až do roku 1982, kdy se na Vysoké škole báňské začala realizovat myšlenka vytvořit informační systém, který by se měl svou funkcionalitou stát elektronickým učitelem studentů a pomocníkem pedagogů [18]. Od té doby uplynulo již několik desítek let a LMS Barborka se vyvíjí i nadále v roce 2013. Během těchto let byl systém vyvíjen a zlepšován také hlavně díky mnoha vědeckým a studentským pracím, které byly součástí vývoje. Ty pojednávaly o různých důležitých tématech a oblastech e-learningu, aby co nejlépe mohly zdokonalovat proces výuky studentů a vytvořit tak co nejlepší teoretickou základnu pro tvorbu LMS. Podstatou bylo zkoumání a následná analýza vyučovacích stylů učitelů, jejich používané metody a individuální odlišnosti u různých studentů.

V posledních deseti letech byly dle současného trendu na systém kladeny požadavky pro veřejnou dostupnost na internetu, tudíž se od roku 2002 začal vyvíjet jako webová aplikace typu klient-server síťové architektury a systém byl označen verzí 1.0 [2] [18]. O rok později po vytvoření funkční verze se tento LMS začal používat v akademické sféře na různých vysokých školách, i ve sféře komerční, jelikož plně poskytoval všechny funkce z oblasti e-learningového vzdělávaní [19]. Projekt Barborka poté získal také několik ocenění, například první cenu na mezinárodní konferenci ICETA[3] 2003 [20]. Další verze [4] vyvinuta v roce 2006 obsahovala novou modulární architekturu a platformu pro praktické experimenty v oblasti inteligentního adaptivního učení [20]. Tato verze ovšem obsahovala jen schopnost systému reagovat na aktuální znalosti studenta a nezohledňovala vlastnosti studenta, jeho učební styly a autoadaptaci výuky [19]. Požadované funkcionality byly v roce 2010 obsaženy v další vytvořené verzi systému [5].

3.2 Vývoj nové verze

V současné době je vytvořena a úspěšně realizována čtvrtá verze LMS Barborka, což je mimojiné i cílem této diplomové práce, respektive implementace samotného řídicího systému, studentského a administrátorského subsystému. Vývoj probíhal v týmu, který byl složen z metodiků a programátorů, kde každý programátor měl z větší části na starost implementaci svého modulu.

Systém je naimplementován ve skriptovacím programovacím jazyce PHP 5.3.8 využívající opensource[6] framework Nette 2.0, Javascript, JQuery v1.7.2 a pracuje s databází MySQL 5.1 pomocí databázové vrstvy dibi 2.0.1 a Nette\Database. Pro formátování obsahu je použita knihovna pro dynamicky psané kaskádové styly LESS. Výběr programového jazyka PHP a databáze MySQL byl na základě dobrých zkušeností s těmito technologiemi z vývoje předešlých verzí a také na základě výhod jaké mají. Což je například vývoj software bez nutnosti placení jakýchkoliv poplatků či bez žádostí o povolení použí-

[2] První verze systému je dostupná na webu http://barborka.vsb.cz/lms/

[3] Mezinárodní konference o nově vznikajících technologiích a aplikacích e-learningu

[4] Druhá verze systému je rovněž dostupná na http://barborka.vsb.cz/ barborka2/

[5] Třetí verze se nachází na http://barborka.vsb.cz/barborka3/

[6] Počítačový software s otevřeným zdrojovým kódem

vat technologie kvůli licencím [21]. Dále, že jsou tyto technologie jednoduché na použití, velmi rozšířené a většina webových serverů je podporuje. Kvůli zrychlení a zkvalitnění vývoje systému byl použit framework pro vývoj webových aplikací Nette. Už podle slov autora tohoto moderního frameworku *„... v otázce bezpečnosti a ladění chyb Nette vážně nemá konkurenci, ...“*[22] jsou důvody použití zřejmé. Ty jsou rychlost aplikace, bezpečnost aplikace, pohodlné a přehledné ladění chyb, architektura MVC a mnoho dalších funkcionalit, jako je automatické generování formulářů s validací, lokalizace, přehledná práce se session, DI či konfigurace prostředí [23]. Další výhoda je opět svobodná licenční politika [24].

Používá se objektově orientovaný přístup vývoje software. Vývoj probíhal v iteracích, kdy po dokončení implementace a otestování jedné části se začaly analyzovat a získávat nové požadavky pro implementaci další části systému. Základní techniky sběru požadavků a zajištění správné funkcionality byly prototypování a diskuze s metodiky. Místo toho, aby se implementovala ostrá verze ihned, vytvářely se prototypy metod, přičemž se vlastně jednalo o zobrazení a simulaci funkčností pro uživatele bez naimplementované logiky. Vstupní a výstupní parametry byly u prototypů vloženy přímo do kódu. Každá vytvořená část a každý případ užití prototypu systému se diskutovaly. Vedle reálné ostré verze tedy existuje ještě jedna nefunkční prototypová.

3.2.1 Důvody tvorby systému

Oproti předchozí verzi, která byla jen rozšířením a doplněním modulů verze druhé, byla tato verze vyvíjena znovu od začátku, tedy od samotného jádra systému. Hlavní důvody tvorby nového LMS jsou (bez uspořádání dle priority):

Nové požadavky
Existuje teorie adaptivní výuky, která není dosud realizována v žádném LMS. Oproti předchozí verzi došlo k úpravám pro celou tuto individualizovanou adaptivní výuku a její nové automatizované řízení (mimojiné i ke změně autorské databázové struktury a s tím spojenými změnami v datové a aplikační vrstvě). S tímto souvisí potřeba nových funkcí pro rozpoznání učebního stylu studenta, určení jeho optimálního výukového stylu, přidělení upraveného výukového materiálu a provázení výukou studenta. Dále se změnila vnitřní struktura subsystémů, přidala se nová logika administrace předmětů, výuky a expertních funkcí. Některé funkce již spadají pod jinou roli než předtím. Také přibyly nové požadavky pro celkovou strukturu (např. požadavky pro dynamické menu a oprávnění). Nakonec již nebylo potřeba několik nevyžitých částí systému (např. některé komunikační nástroje).

Zlidštění uživatelského rozhraní
Podle reakcí uživatelů nebylo uživatelské rozhraní předchozí verze systému dostatečné uživatelsky přívětivé. Uživatelé měli problém se zorientovat v aktuální obrazovce, kde se zrovna nacházeli, a ve formulářích, které neměly popsané jednotlivé položky. Nakonec negativně působilo i kontrastní pozadí. Proto bylo potřeba vytvořit grafické rozhraní, které zvýrazňuje aktuální zvolenou elementární funkci v nabídce, která se nachází vždy v levé části obrazovky, a zobrazuje se vždy stejným způsobem ve stejném grafickém

formátu. U zobrazení každé této funkce se zobrazují titulky shodné s odkazem nabídky a formuláře obsahují jen ty nejnutnější prvky se srozumitelným popisem obsahu.

Sjednoceni prvků uživatelského rozhraní
Části uživatelského rozhraní, které vykonávaly stejnou akci (zobrazení nabídky, tlačítka či hlášení) se zobrazovaly v různém formátu. Dokonce se například stávalo, že v jednu chvíli se na jedné obrazovce vyskytovaly tři různé typy tlačítek, které měly různý vzhled. Dále také různá informační a uživatelská hlášení vzbuzovala nesourodost systému. Proto bylo potřeba vytvořit jednotné šablony pro tyto prvky grafického rozhraní.

Sjednocení postupů a zpřehledněni kódu
Kvůli novým potřebným úpravám, změnám a zásahům do cizího kódu bylo nutné oproti předchozí verzi více sjednotit postupy vývoje všech částí systému a obecně zpřehlednit zdrojový kód. Obecně se dá říci, že se jednalo o potřebu použití novějších technologií. To bylo dosaženo zavedením několika metodik, jako je OOP s dostačujícími komentáři metod a tříd, vytvořením definované struktury použitého frameworku, který podporuje vývoj subsystémů a rozděluje jednotlivé soubory do přehledné adresářové a logické struktury. Dále také dohodou na jednotném jazyku a konvenci pojmenování souborů, tříd, modulů, metod a tabulek, vytvářením znovupoužitelných systémových komponent a používáním stejných Javascript a JQuery komponent.

3.2.2 Vývojové nástroje

Pro zkvalitnění samotného vývoje se používaly různé podpůrné nástroje. Vývojové prostředí bylo zvoleno opensource projekt IDE Netbeans 7.0. Pro zprovoznění Nette frameworku v tomto prostředí bylo potřeba nainstalovat plugin, který mimojiné způsobuje zvýraznění syntaxe, doplňování kódu, generování šablon nebo kódu pro komponenty včetně formulářů, rozumí architektuře MVC, rozpoznává anotace atd. [25]. Pro základní administraci databáze ve webovém rozhraní se používal nástroj phpMyAdmin.

Další použitý nástroj pro verzování a administraci zdrojových kódů byl opensource systém SVN, jenž zvolené vývojové prostředí automaticky podporuje. Zprostředkovává zobrazení změn v kódu, návrat k předchozím commitovaným verzím či částem kódu. Také je pomocí něho možné mít zdrojový kód plně pod kontrolou a jednoduše lze zjistit, kdo kdy provedl jaké změny. Dále byl také použit nástroj PHPUnit pro jednotkové testování. Jednotkové testy jsou také spouštěny při sestavování ostré verze.

Posledním použitým nástrojem byl webový systém Trac, což je bug tracking systém pro řízení projektu, správu a sledování chyb při práci všech vývojářů. Ten obsahuje informace o celkovém vývoji včetně odkazů na pomocné nástroje. Jsou v něm také zaznamenávány zápisy ze schůzí, vytvořené programátorské manuály a návody. Další hlavní funkce Trac systému je plán práce, správa chyb a úkolů, kde si každý programátor může vytvořit hlášení (ticket) s detailní specifikací problému, chyby či úkolu a přidělit ho jinému programátorovi pro vyřešení. Tato hlášení se automaticky odesílají na jeho email. Nakonec je v Trac obsažen externí proces sestavování prototypové verze i aktuální ostré verze ze souborů na SVN.

3.3 Subsystémy a jejich funkcionality

LMS Barborka 4 se skládá z řídicího systému a z pěti hlavních subsystémů. Řídicí systém zahrnuje základní funkcionalitu, zprostředkovává vykonávání funkcionalit ostatních modulů. Takové rozdělení na jednotlivé části je vytvořeno podle definice a podle existence pěti typů uživatelů, kteří s Barborkou pracují. Hlavní subsystémy obsahují jednotlivé elementární oprávněných uživatelů (role student, tutor, autor, expert, administrátor).

Adresářová struktura
Systém využívá strukturu MVC a používá návrh rozložení celku na moduly a danou adresářovou strukturu podle frameworku Nette[7]. V adresářové struktuře se nachází všechny používané knihovny, zdrojové kódy, šablony, obrázky, základní zaváděcí program a soubor s konfigurací parametrů systému a parametrů pro připojení k databázi. Dále to jsou javasctipt soubory, less soubory a jiné soubory, jenž jsou potřebné pro správný běh frameworku. Obsahem této struktury nejsou jednotkové testy, prototypová verze, soubory pro sestavení ostré verze, databázové soubory a dokumentace.

Řídicí systém
Hlavní část je implementována v modulu označeném jako BaseModule. Řídicí systém také obsahuje modul zahrnující funkce veřejně přístupné a modul s funkcemi, které jsou společné pro všechny oprávněné uživatele. Jádro je složeno z částí, bez kterých by nemohla být zajištěna základní funkčnost řídicího systému. Tyto části obsahují metody pro oprávnění a kontrolu práv (autentifikace, autorizace), pro práci s globálními proměnnými systému či s uživatelským nastavením. Obsahují také metody pro generování dynamického menu společné s obecnou komponentou pro generování menu, protokolování akcí uživatele, kompletní správu všech souborů, metody pro práci se slovníkem a jazykovou lokalizací.

Dále jsou součástí jádra dvě hlavní systémové komponenty. Komponenta pro společnou tabulku, jež vypisuje celé seznamy, a se kterou lze na datech provádět kromě operací CRUD i dílčí akce pro specifické případy. S ní je spojena druhá pro práci s globálními filtry, podle které se filtrují záznamy v zobrazených tabulkách. Kromě hlavních komponent existují i další společné systémové komponenty pro zobrazování vlastních formulářových prvků, odpočítávání času či diskuze mezi uživateli. Nakonec se zde vyskytují i jQuery komponenty pro kalendář, přetahování hodnot mezi dvěma sloupci (multiselect) a CKeditor.

Modul obsahující veřejné funkce
Kromě funkcí pro registrované uživatele systém obecně obsahuje také funkce dostupné všem uživatelům internetu. To je základní proces pro registraci a přihlášení do systému. Tento proces je vytvořen tak, aby mohl pracovat s protokolem LDAP místní školní organizace, načítat a ověřovat tak uživatele z databáze školy. Další součástí jsou metody zpracovávající ukládání,omezené stahování a zobrazovaní všech souborů v systému. Poslední nedílnou součástí tohoto modulu je zobrazování chybových a informačních hlášek uživatelům v případě chyby či nestandardního chování systému. Tuto funkcionalitu obsahuje modul s názvem PublicModule.

[7] Rozložení na moduly je ukázáno na stránce http://doc.nette.org/cs/presenters#toc-moduly

Modul obsahující společné funkce
Všechny metody vykonávající elementární funkce, ke kterým mají přístup uživatelé všech rolí systému, se nachází v modulu pojmenovaném jako OstatniModule. V tomto modulu se zatím vyskytují jen funkce pro změnu osobních uživatelských údajů a pro změnu hesla.

Subsystém Student
Tento subsystém je naimplementován v modulu s názvem StudentModule a skládá se z funkcionalit určených pro všechny studenty. Což je nejprve zobrazení dotazníků a uložení jejich učebních charakteristik Dále je to správa všech dostupných předmětů s detailními informacemi, což znamená předměty veřejné dostupné všem nebo aktuální studentem zapsané. Student má možnost se k předmětu zapsat s aktuálním heslem předmětu. Hlavní část tohoto subsystému tvoří adaptivní výuka, která obsahuje hierarchickou navigaci předmětů, navigaci variant včetně hloubek pomocí tabulky - dvanáctice a zobrazení samotných výukových vrstev či testovacích vrstev. Zobrazuje se ovšem jen obsah vrstev, to znamená, obsah komponent dané vrstvy. Podle toho, jak se student učí, se virtuálním učitelem obměňuje výuka. Během výuky se provádí testování studentů a zobrazují se testovací otázky, na které student odpovídá. Reakci k odpovědím generuje opět virtuální učitel. Systém si pamatuje poslední stav výuky (zobrazené rámce a vrstvy, studentem zvolené varianty) a všechny vyplněné odpovědi. Při následujícím sezení studenta se automaticky načte tento poslední stav. Pro zobrazení vrstev a navigační n-tice slouží vytvořené komponenty.

Další důležitá část modulu Student je testování znalostí studentů. Ti si mohou opětovně spouštět autotesty na daný čas a po ukončení procházet vyhodnocení. Také se mohou zapisovat k termínům ostrých testů, prohlížet své již vykonané ostré testy anebo tyto testy spouštět. Vyhodnocení odpovědí je vykonáváno metodami modulu tutor.

Nakonec si studenti mohou prohlížet zadané úkoly a odevzdávat vypracované řešení, přičemž mohou elektronicky diskutovat s vyučujícím.

Subsystém Tutor
Subsystém se vyskytuje v modulu TutorModule. V něm se nachází administrace všech záznamů a aktivit spojených s organizací výuky učitele – tutora. Jedná se o přehled a správu těch předmětů včetně hesel, které učitel tutoruje. Může také přidělovat ke svým předmětům ostatní učitelé. Dále se zde vyskytuje přehled všech studentů zapsaných k předmětům, přičemž tutor sám může také studenty zapsat.

Funkce související s administrací aktivit jsou ovšem hlavní součástí tohoto subsystému. Mezi aktivity zatím patří úkoly, autotesty a ostré testy. K předmětům je možné vytvářet všechny tyto aktivity a spravovat je. Testům a autotestům se musí vytvořit struktura. Poté jsou složitými algoritmy testy vygenerovány a můžou být zobrazeny studentům. Tento modul také obsahuje funkcionalitu vyhodnocení studentových odpovědí, používá zobrazení vyhodnocení ze studentského modulu a umožňuje export jednotlivých vyhodnocených testů. Systém také umožňuje vytvářet či spravovat termíny na ostré testy a zapisovat studenty k těmto vypsaným termínům. Nakonec se v modulu Tutor vyskytuje ještě administrace všech školských číselníků (tzn. číselníků obsahující akademické roky, obory, fakulty, katedry, školy a typy studia).

Subsystém Autor
Smyslem subsystému autor, implementovaném v modulu AutorModule, je vytváření, zařazování a vzájemné propojování dílčích částí adaptivních výukových opor pomocí propracovaných formulářů. Elementární funkce, jež v tomto modulu existují, kompletně spravují všechny části výuky: předměty, kapitoly, lekce, rámce, varianty, vrstvy, testovací vrstvy, komponenty a odpovědi.

Subsystém Expert
Tento subsystém řeší hlavní funkcionalitu činnosti virtuálního učitele, včetně načtení informací o studentovi a o struktuře výukových materiálů, určení optimálního řízení výuky studentovi na míru, adaptivní změnu výuky podle aktuálního učebního stylu studenta a definování obecných pravidel pro celkové zobrazení výuky. Modul pracuje s aktuálními znalostmi a učebními charakteristikami studentů. Dále obsahuje generování reakce na studentovy odpovědi testovacích otázek při výuce. Součástí jsou taktéž elementární funkce pro zprovoznění a tvorbu studentských dotazníků (otázek a odpovědí), správu měřených vlastností studentů a výsledků vyplněných dotazníků.

Další důležitou částí je modelování výuky a s tím spojená správa virtuálních studentů. Nejdříve je vytvořena modelová virtuální opora, poté jsou zvoleni virtuální studenti a podle těchto vstupů je posléze vykreslen graf modelu výuky, zobrazující průběh výuky každého studenta. Nakonec jsou zde obsaženy také metody pro analýzu a data-mining dat, získaných z protokolu uloženého v databázi, tedy z akcí a hodnocení studentů. To vše se nachází v modulu pojmenovaném ExpertModule.

Subsystém Administrátor
Poslední subsystém zaštiťuje funkcionalitu administrace celého systému, jeho třídy se nacházejí v modulu AdminModule. Administrátoři se starají o správu všech uživatelů systémů. Mohou také do systému importovat větší počet uživatelů najednou. Další nedílnou součástí je administrace zpřístupnění registrace studentů do systému pod heslem a dynamická tvorba struktury menu. Ta obsahuje administraci všech elementárních funkcí systému, využívaná také pro přístup k funkcím a pro oprávnění jednotlivých uživatelů. Mezi administrační funkce také patří správa systémového slovníku, typů slovních spojení, globálních systémových proměnných a zobrazení celkové struktury databáze (model databáze).

4 Vývoj LMS

V této části se nachází popis dosavadního životního cyklu systému Barborka 4 s detailně rozbranými jednotlivými etapami vývoje projektu.

4.1 Zadání a specifikace požadavků

Obecným zadáním projektu je návrh a implementace řídicího systému, studentského a administrátorského modulu LMS Barborka 4. Systém musí být navržen a implementován se vší funkcionalitou a rozvržením subsystémů dle teoretického základu představující adaptivní výuku. Návrh a implementace musí být provedena ve spolupráci s ostatními programátory podílejícími se na vývoji toho systému či ostatních modulů. Pro budoucí provoz je potřeba vytvořit programátorskou příručku a uživatelský manuál.

4.1.1 Funkční požadavky

Proč nový systém?
Hlavní důvod, proč vytvořit nový systém je ten, že výše zmíněná teorie adaptivní výuky není ještě realizována v žádném existujícím LMS. Další důvody pro tvorbu nového systému jsou detailně popsány v kapitole 3.2.1.

K čemu má systém sloužit?
V první řadě systém slouží k výzkumu personalizované výuky s cílem vytvořit funkčního inteligentního virtuálního učitele s automatickou adaptací výukového procesu. S tímto jsou spojené expertní činnosti určení učebního stylu, výukového stylu a formulace pravidel, jak je definováno v teoretickém základu. Dále jsou s výzkumem spjaty analýzy pomocí statistických a data-miningových metod.

Sekundárně systém slouží k plnohodnotné e-learningové výuce studentů, tedy musí obsahovat všechny činnosti v rámci e-learningu v kapitole 2.1. Taktéž musí obsahovat evidence studentů, výukových materiálů, administraci studia a studentských aktivit.

Prioritizace cílů:

1. Implementace adaptivní výuky a všech přidružených funkcionalit
2. Přidání studentských aktivity jako doplňky k e-learningové výuce
3. Přidání doplňků k evidenci studentů, výuky, skupin, aktivit aj.

Kdo se systémem pracuje?
Se systémem bude pracovat šest různých typů uživatelů s pěti přístupovými právy.
1.) Neregistrovaní uživatelé se budou nejspíše chtít registrovat do systému příležitostně.
2.) Studenti středních a vysokých škol, kteří budou se systémem pracovat nejčastěji. Předtím, než budou pracovat s tímto systémem, měli by být obeznámeni s uživatelskou příručkou popisující možnosti systému, ovládáním výuky, formu zobrazení testovacích otázek u výuky nebo testů včetně vyhodnocení. Měli by znát rozdíl mezi autotesty a ostrými testy.

3.) Vysokoškolští a středoškolští pedagogové, kteří budou spravovat a produkovat adaptivní opory podle existující metodiky tvorby personalizovaných e-learningových opor. Obecně autoři přistupují do systému zřídka. Častěji tedy jen při vytváření předmětu a po vytvoření předmětu už opory jen upravují či doplňují.
4.) Dalšími uživateli budou taktéž pedagogové, ale v jiné roli. Tito uživatelé se budou starat o evidenci studijních záležitostí, správu testů, skupin a hodnocení studentů. Předpokládá se, že tito tutoři budou využívat systém často.
5.) Experti, kteří budou tvořit dotazníky a pravidla, používat analytické a expertní funkce systému. Experti budou funkce systému využívat zřídka.
6.) Administrátoři systému budou mít na starost základní chod a správu celého systému. V tomto případě se vlastně jedná o vývojáře sytému, kteří mají přístup k databázi a používají administrátorské funkce. Jejich přístup bude během ostrého finálního provozu ze začátku častý a po následném období optimalizace a opravování chyb bude spíše příležitostný.

Podle zadání se budou následující analýzy věnovat řídicímu systému, rolím student, administrátor a k nim přiřazeným modulům. Výjimkou je správa virtuálních studentů a zobrazení modelu výuky v modulu expert.

Vstupy do systému
Modul Public
- Ověření přihlašovacích údajů
    - povinné atributy: přihlašovací login, heslo
- Registrace nového uživatele
    - povinné atributy: přihlašovací login, heslo, heslo pro potvrzení, jméno, příjmení, email, heslo pro registraci
    - nepovinné atributy: titul, datum narození, id školy, id fakulty, id studijního oboru, ročník

Modul Administrátor
- Vložení uživatele
    - povinné atributy: login, heslo nebo binární atribut určující přihlašování přes LDAP, jméno, příjmení, email, číselné id rolí uživatele, hlavní role
    - nepovinné atributy: titul, datum narození, dodatečná jemná práva, id školy, id fakulty, id studijního oboru, id katedry, ročník, místnost, katedra
- Import uživatelů z CSV souboru
    - povinné atributy: login, heslo nebo binární atribut pro LDAP, jméno, příjmení, email
    - nepovinné atributy: titul, datum narození, zkratka školy, zkratka fakulty, zkratka studijního oboru, zkratka katedry, ročník, místnost, katedra
- Vložení zpřístupnění registrace do systému
    - povinné atributy: datum a čas začátku a konce zpřístupnění, registrační heslo

- Vložení položky menu
    - povinné atributy: zobrazovaný text v menu, id rodičovského uzlu, identifikátor příslušné elementární funkce, pořadí položky
    - nepovinné atributy: odkaz položky, binární atribut určující zobrazení položky v menu, popis, nápověda
- Vložení typu slovního spojeni
    - povinné atributy: unikátní klíč typu slovního spojení, popis
- Vložení fráze do slovníku
    - povinné atributy: unikátní klíč slovního spojení (může být přímo český výraz), anglický výraz, id typu slovního spojení
    - nepovinné atributy: český výraz
- Vložení globální proměnné systému
    - povinné atributy: název, hodnota
    - nepovinné atributy: id modulu

Modul Student
- Uložení vyplněné odpovědi u testovací otázky
    - povinné atributy: identifikátory zvolených variantních odpovědí, text tvořených odpovědí, pole se slovy pro doplňovačku
- Vložení příspěvku k úkolu
    - povinné atributy: text příspěvku nebo soubor pro nahrání do systému

Modul Ostatní
- Vložení údajů
    - povinné atributy: email
    - nepovinné atributy: titul, datum narození, id školy, id fakulty, id katedry, ročník, místnost, telefon
- Vložené nového hesla
    - povinné atributy: staré heslo, nové heslo, nové heslo pro potvrzení

Část modulu Expert
- Vložení virtuálního studenta
    - povinné atributy: login, varianty smyslového vnímání (verbální, vizuální, auditivní, kinestetické), sociální aspekty, afektivní aspekty, systematičnost, teoretické odvozování, experimentování, detailistický a holistický postup učení, pojetí učení, autoregulace, úspěšnost

Výstupy ze systému
Modul Public
- Zobrazení chybových hlášení a upozornění

Modul Administrátor
- Seznam uživatelů (+ detail a editace uživatele)

- Seznam aktuálních zpřístupnění (+ editace aktuálního zpřístupnění)
- Historie zpřístupnění
- Seznam podpoložek vybrané položky menu (+ editace zvolené položky)
- Seznam slovních spojeni (+ editace spojení)
- Seznam typů slovních spojeni (+ editace typu)
- Seznam globálních proměnných (+ editace proměnné)
- Seznam databázových tabulek, seznam atributů databázové tabulky

Modul Student
- Seznam zapsaných předmětů studenta v semestru (+ detail zapsaného předmětu)
- Seznam veřejných předmětů aktuálního semestru (+ detail veřejného předmětu)
- Seznam dostupných předmětů k zápisu (+ detail dostupného předmětu)
- Zobrazení výuky:
    - hierarchická struktura výuky pro navigaci
    - seznam aktuálních možných variant (12tice)
    - seznam vrstev výuky (komponent) a testovacích vrstev (otázky s odpověďmi)
- Seznam úkolů předmětu
- Detail úkolu:
    - zadání úkolu (komponenty)
    - seznam příspěvků a odevzdaných souborů
    - vložení příspěvku
- Seznam dostupných autotestů
- Seznam ostrých testů dostupných ke spuštění
- Seznam dostupných terminů testů k přihlášení
- Seznam všech testů studenta (ukončené, aktuální a budoucí)
- Zobrazení pokynů (komponent) autotestu či testu
- Zobrazení autotestu či testu:
    - vždy je zobrazována hlavička testu
    - po jedné otázce v novém okně – zobrazená jedna otázka (komponenty) a příslušné odpovědi
    - všechny otázky najednou v původním okně - zobrazené otázky (komponenty) a příslušné odpovědi najednou
- Zobrazení vyhodnocení autotestů či testů
    - hlavička vyhodnocení testu
    - seznam otázek (komponent) a odpovědí s bodovým ohodnocením
- Detail testu:
    - hlavička testu
    - seznam otázek (komponent) a odpovědí stejně jako u zobrazení vyhodnocení

Modul Ostatní
- Editace údajů
- Editace hesla

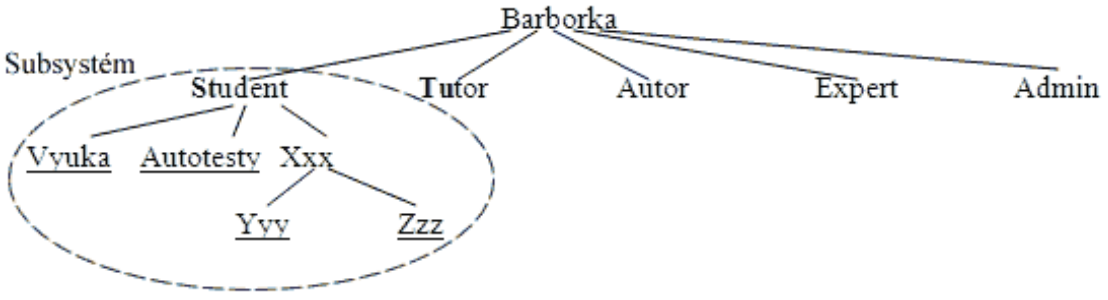
Řídící systém
- Zobrazení uživatelského rozhraní
    - zobrazení menu
    - zobrazení dostupných jazyků
    - zobrazení společných komponent

Část modulu Expert
- Seznam virtuálních studentů (+ detail a editace virtuálního studenta)
- Seznam virtuálních studentů s vlastnostmi
- Výběr opory
- Zobrazení grafu modelu výuky

Seznam funkcí
Základní funkční požadavky získané ze zadání byly tyto:

Generátor menu
Pro generování menu je použita stromová hierarchie položek (obrázek 4) uložená v databázi a ta tvoří skelet řídicího programu. Stromová hierarchie je složena z elementárních funkcí podle úrovní DFD. Menu se generuje dynamicky, tedy se nemění program, ale pouze data. Administrátor má možnost jakkoliv měnit strukturu menu, doplňovat či vyměňovat elementární funkce.

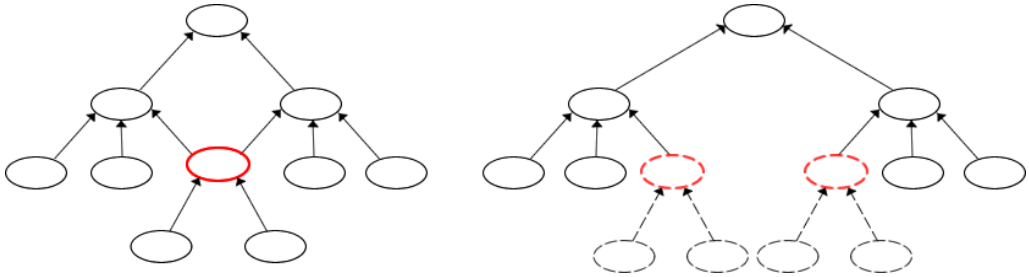
Pro stromovou strukturu platí určitá pravidla:
- Listy jsou elementární funkce.
- Uzly jsou skupiny elementárních funkcí.
- Základní větve jsou jednotlivé subsystémy.
- Každý list má vazbu na 1 elementární funkci.
- Dvě různé položky (listy) můžou mít vazbu na 1 společnou elementární funkci, tyto funkce jsou ovšem pojmenovány různě.
- Každý uzel má odkaz na funkci svého následovníka či svoji vlastní elementární funkci.
- Kořen struktury zahrnuje samotný systém – v databázi je ale označen jako null.
- Elementární funkce (list) může i nemusí být zobrazena v menu.



Obrázek 4: Strom elementárních funkcí

Společné funkce
Všechny pro moduly společné funkce se nachází ve zvláštním modulu. Elementární funkce mají odlišný prefix elementární funkce (označení modulu a číselné označení), ale odkaz na funkci budou mít stejný.

Podle teoretického návrhu se však jedná o společnou elementární funkci, která se odkazuje na více rodičů, to porušuje stromovou strukturu a vytváří se tím cyklus. Tento problém byl vyřešen právě vytvořením různých potomků, kteří se pouze odkazují na společnou elementární funkci. Tato situace je popsána na obrázcích 5:



Obrázek 5: Společné menu

Oprávnění
S hierarchickou strukturou elementárních funkcí úzce souvisí oprávnění uživatelů. Každému uživateli jsou přiděleny role a jemná práva (podmnožina identifikátorů elementárních funkcí). Podle tohoto se kontroluje přístup k elementárním funkcím daných větví této stromové struktury.

Pro oprávnění platí taktéž jistá pravidla:
- Uživatel s danou rolí má přístup ke všem elementárním funkcím příslušného modulu.
- Uživatel má přístup k elementární funkci či skupině elementárních funkcí modulu, který přísluší jiné roli právě tehdy, pokud má identifikátor daného listu či uzlu ve svých jemných právech.
- Pokud má uživatel v jemných právech uzlový identifikátor, pak má přístup také ke všem jeho potomkovským funkcím.
- Uživatel může mít v jednom řetězci přístup jen k jedné akci elementární funkce.
- Odkaz a identifikátor elementární funkce z requestu musí korespondovat s odkazem a identifikátorem funkce uložené v databázi.
- K veřejně dostupným funkcím modulu public mají přístup všichni uživatelé.
- Kontrola přístupnosti se provádí postupným porovnáním všech částí identifikátorů.

Evidence uživatelů
Pro evidenci uživatelů a jejich práv se používá jedna databázová tabulka sy_uziv. Jelikož systém pracuje zvlášť se studenty a zvlášť se zaměstnanci, jsou vytvořeny speciální tabulky st_student a sy_zamestnanec, obsahující konkrétní rozšiřující údaje pro danou skupinu uživatelů.

Slovník
Systém umí provádět překlad systémových hlášek, názvů, textů a frází do cizího jazyka (angličtiny). Jednoduše lze přepnout na cizojazyčnou verzi. Pro rozdělení textů, které se mají překládat, existuje číselník s typy slovních spojení. Klíčem může být zvolený unikátní řetězec, pro který existuje příslušný český i anglický výraz, ale taky klíčem může být přímo český výraz, pro který existuje jen příslušný anglický překlad.

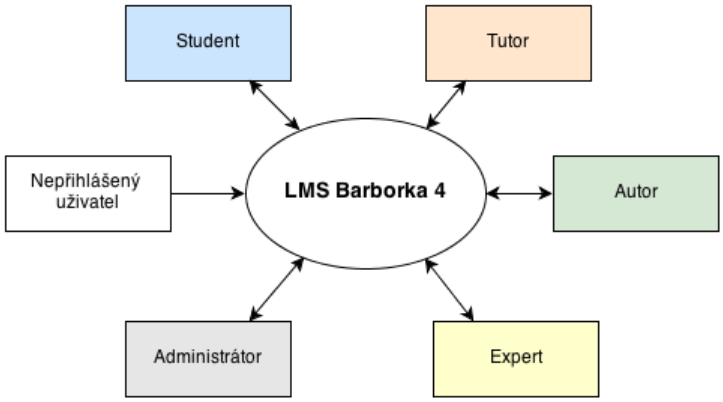
Osobní prostředí
Pro přihlášeného studenta si systém pamatuje poslední konfiguraci jeho osobního prostředí. Ukládá se stav výuky, testů a globálních filtrů. Ve výuce jsou ukládány všechny zobrazené vrstvy jednotlivých rámců a stavy testovacích vrstev (stavy otázek s vyplněnými odpověďmi). Taktéž jsou ukládány aktuální spuštěné testy a autotesty s obsahem studentových odpovědí. Nakonec jsou ukládány poslední konfigurace globálních filtrů.

Systémový katalog
Pro roli administrátor je potřeba zobrazovat aktuální stav databáze tzv. „model databáze“, který zobrazuje seznam všech tabulek. V detailu tabulky se nachází seznam atributů s referencemi na ostatní tabulky, lineární zápis a závislé tabulky.

Protokol
Systém umí ukládat do protokolu uživatelem provedené akce s detailním popisem. Je potřeba protokolovat tyto akce:
- Přihlášení do systému – uživatel se přihlásí do systému
- Odhlášení ze systému – uživatel je odhlášen ze systému
- Přihlášení k předmětu – uživatel zvolí předmět
- Přihlášení k lekci – uživatel zvolí lekci, která je jiná než aktuální
- Přihlášení k rámci – uživatel zvolí rámec, jenž je jiný než aktuální
- Přihlášení k variantě – uživatel zvolí variantu jinou než aktuální
- Přihlášení k vrstvě – student zvolí jinou než aktuální nebo odešle testovací vrstvu
- Přihlášení k odpovědi – student odeslal odpověď u testovací otázky
- Přihlášení k testu – student spustil autotest nebo ostrý test

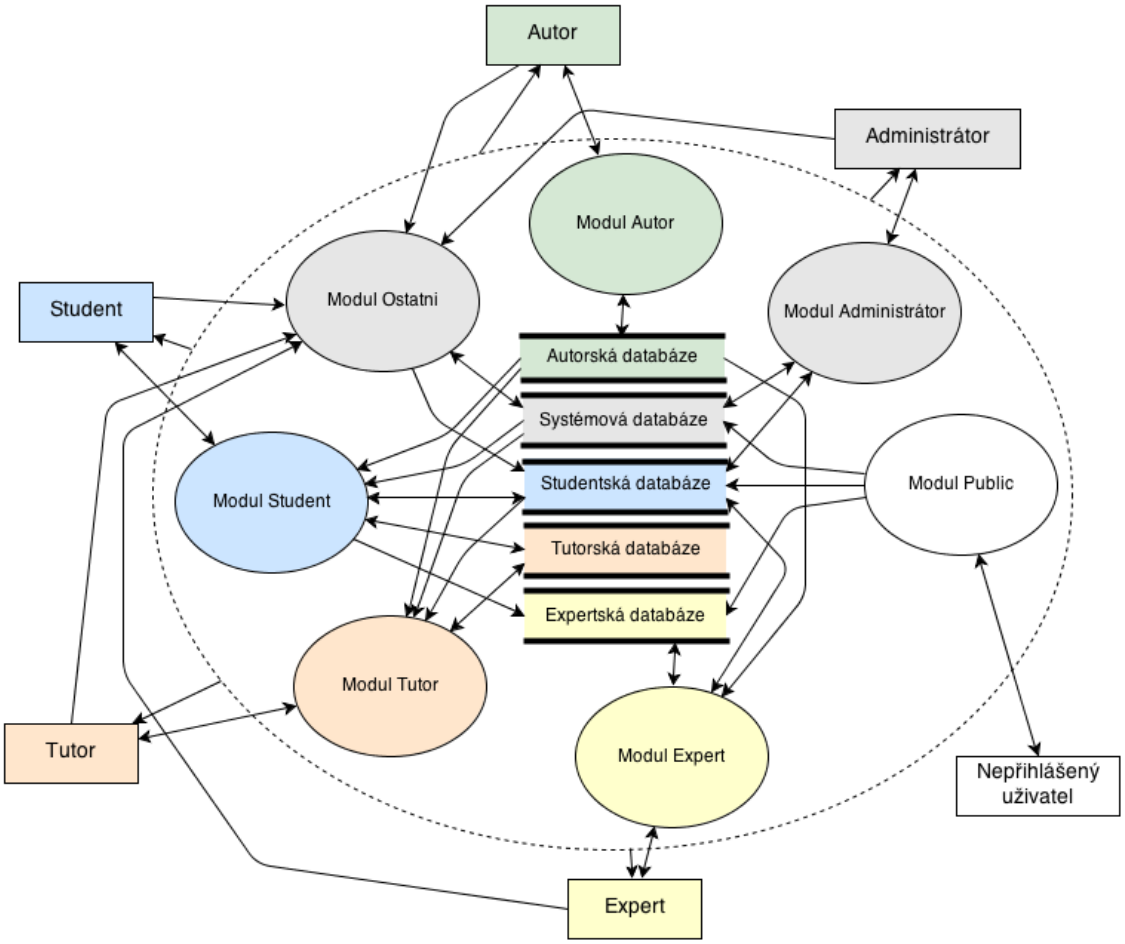
Všechny zbylé elementární funkce vycházejí z požadavků získaných tvorbou prototypů. Seznam všech naimplementovaných elementárních funkcí pro řídicí systém, moduly Student a Administrátor se nacházejí v příloze.

Okolí
Se systémem pracuje 6 různých typů uživatelů v 5 rolích (student, autor, tutor, expert, administrátor) registrovaných uživatelů a neregistrovaní. Vstupní a výstupní datové toky s aktéry jsou ukázány v kontextovém diagramu (obrázek 6)



Obrázek 6: Kontextový diagram

Diagram datových toků 0. úrovně popisující rozdělení systému na moduly (obrázek 7).



Obrázek 7: Diagram datových toků 0. úrovně

4.1.2 Nefunkční požadavky

Na systém je od zadavatele kladeno mimo funkčních i mnoho nefunkčních požadavků:

Pojmenování databázových tabulek a klíčů
Název každé databázové tabulky je složen z prefixu (zkratky) modulu a z části nebo celého názvu tabulky oddělené podtržítkem. Každý modul má svou zkratku složenou ze 2 písmen podle názvu (st, tu, au, ex, sy, pu, os). Zkratka „sy“ zahrnuje systémové i administrátorské tabulky. První znak názvu tabulky mnohdy udává specifickou oblast, pro kterou je tabulka použita (C - číselník, X - vazební tabulka, T - testování, S - student, G - vygenerovaný test, A - aktivita). První čtveřice znaků z názvu každé tabulky je vždy jednoznačná.

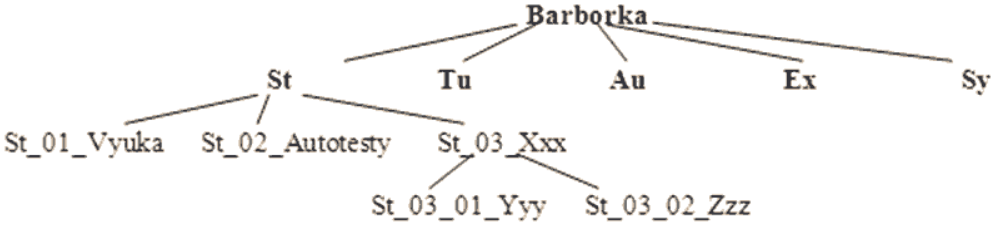
Název identifikátoru (klíče) každé tabulky je složen z prefixu „id_“ a unikátní čtveřice názvu tabulky. Hodnota identifikátorů je složena z 10 znaků, kde první čtyři znaky tvoří taktéž tato unikátní čtveřice a zbylých 6 znaků tvoří číslo doplněné z levé strany nulami.

Vzory:

Název tabulky:	mo_nazevtabulky
Označení identifikátoru:	id_naze
Hodnota identifikátoru:	Naze000000

Systémové proměnné
Každá databázová tabulka obsahuje 3 systémové atributy dat_zap, dat_zme a kdo_zme. Ty obsahují datum vytvoření záznamu, datum poslední změny a identifikátor uživatele, který provedl poslední změnu záznamu.

Pojmenování elementárních funkci
Každá elementární funkce systému je jednoznačně pojmenována. Identifikátor je složen ze zkratky modulu (0. úroveň dle DFD), dvojciferného čísla udávajícího pořadí v aktuální úrovni a z názvu funkce odděleného podtržítkem. Potomkovské funkce mají prefix složený z prefixu funkce rodičovské a nového pořadí dané podúrovně. Vzor pro pojmenování elementárních funkcí je uveden na obrázku 8



Obrázek 8: Pojmenování elementárních funkcí

Syntax
Pro programátory je dáno pravidlo pro pojmenování tříd, proměnných a názvů metod. Pojmenování je založeno na syntaxi programovacího jazyka Java.

Formát základního menu
Menu je potřeba zobrazit v jednoduchém statickém prostředí bez dynamické změny položek, která by dezorientovala uživatele. Menu je složeno ze 2 hlavních části – hlavního menu a levého menu. V menu se zobrazují názvy modulů, funkcí a elementárních funkcí. Položky hlavního menu se nacházejí vedle sebe v horní části obrazovky. Hlavní menu je složeno ze 2 úrovní (0. a 1. úroveň dle DFD), kde nultá úroveň obsahuje pouze moduly.

Levé menu se nachází v levé části obrazovky a položky jsou seřazeny pod sebou. Položky tvoří hierarchii a jsou rozklikávací respektive sbalovací. Při rozkliknutí se pod položkou zobrazí podpoložky další úrovně a při sbalení se schovají. Aktuální položka je zvýrazněna vždy tučně.

Konzistence UI
Grafické rozhraní obsahuje různé kategorie uživatelských prvků, které musí mít vždy stejný formát a vzhled. Jedná se o tlačítka, formuláře, tabulky vypisující seznamy, ale také o chybové a informační hlášení. Tyto všechny kategorie mají sjednocené grafické formáty.

Pro každý modul je vytvořeno jedno barevné schéma s bílým pozadím. Každému tomuto subsystému je pro vykreslení grafických prvků přidělena jiná barva a plochy jsou vyplněny touto barvou ve světlém odstínu.

Architektura
Pro realizaci systému je využita architektura klient – server. Klienti posílají požadavky na server přes internet. Server má přístup k databázi na lokální síti. Systém neumí pracovat v režimu offline bez internetového připojení.

Dokumentace
Při vývoji je potřeba vhodně komentovat všechny potřebné metody a třídy, popisovat rozhraní a psát uživatelskou i programátorskou dokumentaci.

Testovatelnost
Jednotlivé naimplementované metody a komponenty musí být testovatelné, takže v systému je potřeba zavést nástroje pro testování.

Vývojové prostředí
Je požadováno implementovat v jazyce PHP s použitím objektově orientovaného programování. Je potřeba použít MVC framework (doporučen framework Nette). Použití frameworku zprostředkovává konfigurovatelnost a rozšiřitelnost systému. Systém bude pracovat s databází MySQL. Zbylé požadované technologie jsou popsány v kapitole 3.2. a požadované nástroje v kapitole 3.2.2.

Bezpečnost
Systém musí umět autentizovat a autorizovat registrované uživatele a musí umět jim zprostředkovat funkce a data, ke kterým mají přístup. Také musí chránit tato data před zneužitím neautorizovanými osobami.

Optimalizace
Pokud systém v provozu nebude zvládat zátěž a bude odpovídat s dlouhými odezvami, bude potřeba systém optimalizovat na databázové stránce a zprovoznit cachování.

4.2 Analýza

V této kapitole je detailněji popsána datová a funkční analýza systému vzniklá při specifikaci požadavků a funkcí. Jsou použity pouze části modulu Student, Administrátor a řídicího systému.

4.2.1 Datová analýza

Ve studentské části databáze se nachází 3 tabulky a v systémové/administrátorské části je databázových tabulek 12. Tyto tabulky mají vazby na 10 tabulek z tutorské databáze, 3 tabulky z autorské a jednu tabulku z expertní části databáze. Databáze není popsána kompletně, ale jsou popsány pouze tabulky z části student a systém. V ER diagramu jsou vyznačeny i vazby na ostatní tabulky systému, které jsou zobrazeny čárkovaně.

4.2.1.1 Lineární zápis

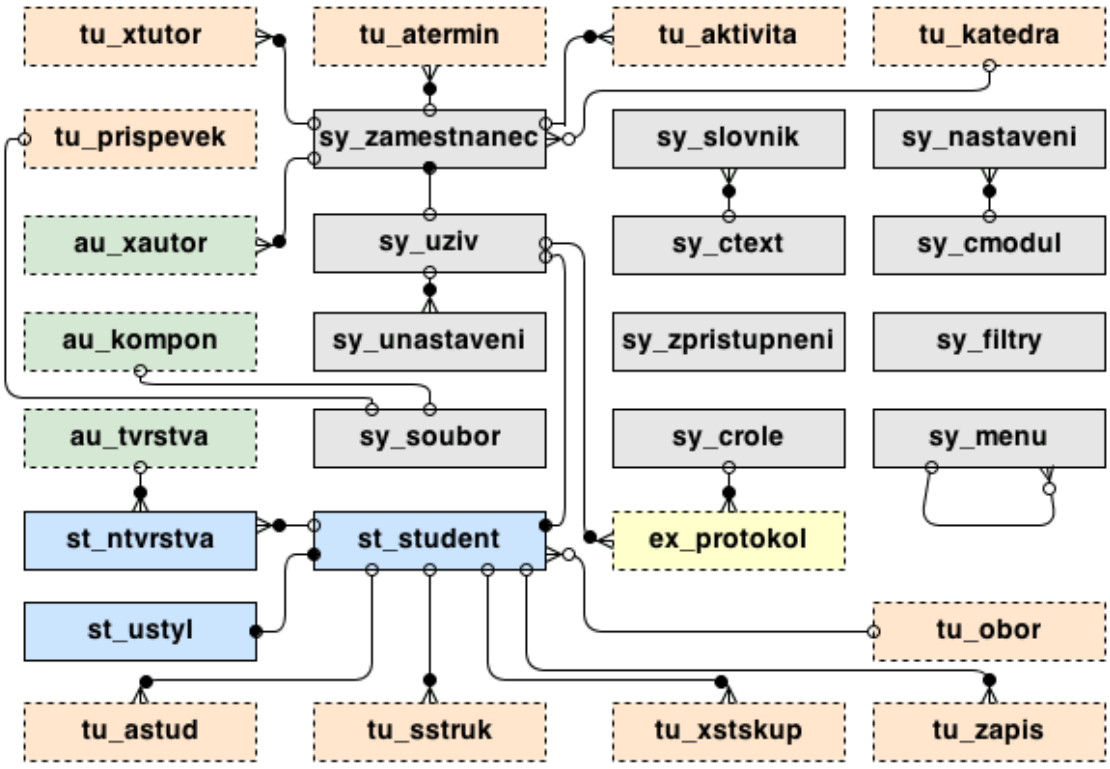
sy_cmodul (id_cmod, nazev_m, dat_zme, dat_zap, kdo_zme)
sy_crole (id_crol, nazev_r, prava_r, dat_zme, dat_zap, kdo_zme)
sy_ctext (id_ctex, zkr_typ, popis_typ, dat_zme, dat_zap, kdo_zme)
sy_filtry (id_filt, filtr)
sy_menu (id_menu, *id_nadm*, iden_m, nazev_m, popis_m, help_m, volani_m, poradi_m, zobrazit_m, dat_zme, dat_zap, kdo_zme)
sy_nastaveni (id_nast, nazev_n, hodnota_n, id_cmod, dat_zme, dat_zap, kdo_zme)
sy_slovnik (id_slov, *id_ctex*, iden_s, nazev_cz, nazev_en, dat_zme, dat_zap, kdo_zme)
sy_soubor (id_soub, nazev, cesta, mime, dat_zme, dat_zap, kdo_zme)
sy_unastaveni (id_unas, *id_uziv*, nazev_n, hodnota_n, dat_zme, dat_zap, kdo_zme)
sy_uziv (id_uziv, login, heslo, ldap, prava, role_admin, role_autor, role_expert, role_student, role_tutor, hlavni_role, dat_zme, dat_zap, kdo_zme)
sy_zamestnanec (*id_uziv*, dat_nar, jmeno, prijmeni, titul, email, *id_katedry*, mistnost, telefon, dat_zme, dat_zap, kdo_zme)
sy_zpristupneni (id_zpri, dat_zme, dat_zap, kdo_zme, zacatek_z, konec_z, heslo_z)
st_ntvrstva (id_ntvr, *id_uziv*, *id_tvrstva*, rezim, stav, odpoved, zprava, dat_zme, dat_zap, kdo_zme)
st_student (*id_uziv*, dat_nar, jmeno, prijmeni, titul, *id_oboru*, rocnik, email, virtualni, dat_zme, dat_zap, kdo_zme)
st_ustyl (*id_uziv*, sver, sviz, saud, skin, stsoc, stafek, stsyst, stteor, stexp, stdetail, sthol, stpoj, stareg, stusp, dat_zme, dat_zap, kdo_zme)

4.2.1.2 Popis vztahů

JE_TUTOREM (sy_zamestnanec, tu_xtutor) 1:N
JE_AUTOREM (sy_zamestnanec, au_xautor) 1:N
VYTVÁŘÍ (sy_zamestnanec, tu_atermin) 1:N
SPRAVUJE (sy_zamestnanec, tu_aktivita) 1:N
UČÍ_NA (sy_zamestnanec, tu_katedra) N:1
ROZŠIŘUJE (sy_zamestnanec, sy_uziv) 1:1

JE_TYPU (sy_slovnik, sy_ctext) N:1
PATŘÍ_POD (sy_nastaveni, sy_cmodul) N:1
SE_UKLÁDÁ_PRO (sy_unastaveni, sy_uziv) N:1
JE_PROTOKOLOVÁN_DO (sy_uziv, ex_protokol) 1:N
VYTVOŘILA_ROLE (ex_protokol, sy_crole) N:1
JE_PŘEDKEM (sy_menu, sy_menu) 1:N
OBSAHUJE (au_kompon, sy_soubor) 1:1
JE_VLOŽEN_V (sy_soubor, tu_prispevek) 1:1
ROZŠIŘUJE (st_student, sy_uziv) 1:1
OBSAHUJE_NASTAVENÍ (st_ntvrstva, au_tvrstva) N:1
OBSAHUJE_NASTAVENÍ_PRO (st_ntvrstva, st_student) N:1
MÁ_DEFINOVANÝ (st_student, st_ustyl) 1:1
STUDUJE (st_student, tu_obor) N:1
JE_ZAPSÁN (st_student, tu_zapis) 1:N
VYKONÁVÁ (st_student, tu_astud) 1:N
VYPLŇUJE (st_student, tu_sstruk) 1:N
JE_PŘIŘAZEN_DO (st_student, tu_xstskup) 1:N

4.2.1.3 ER diagram



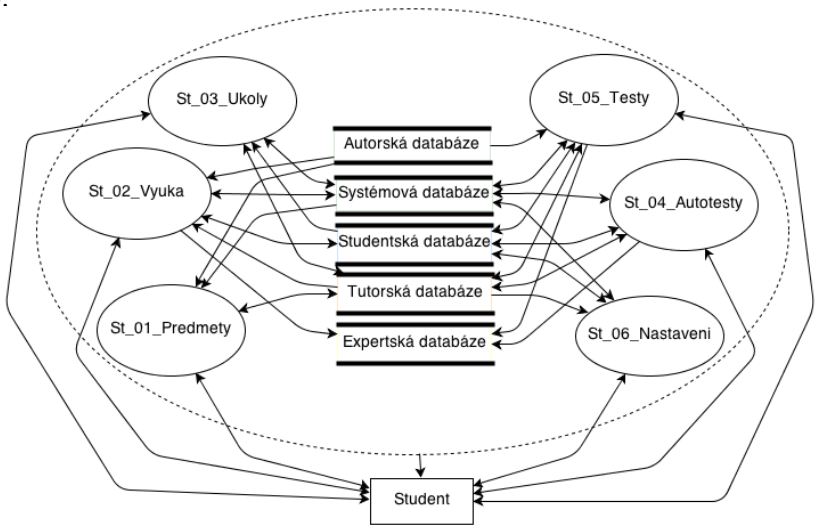
Obrázek 9: ERD

4.2.2 Funkční analýza

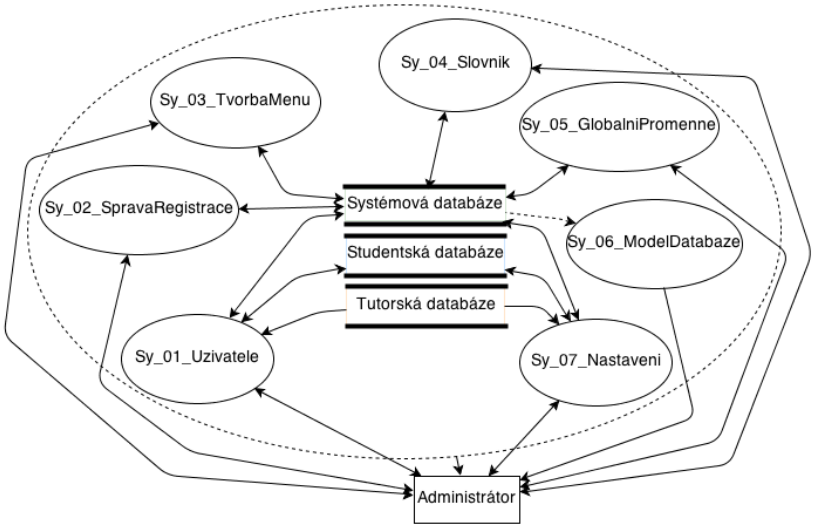
V této části jsou popsány jen vybrané případy užití zaměřující se na zobrazení výuky ve studentském modulu a důležité funkčnosti. Bylo vybráno celkem 25 případů užití a každý use case má svůj unikátní identifikátor uvedený před názvem. Pro detailní popis je u některých uveden sekvenční diagram v příloze či use case diagram.

4.2.2.1 Diagramy datových toků

První jsou zobrazeny DFD 1. úrovně pro moduly Student (obrázek 10) a Administrátor (obrázek 11).

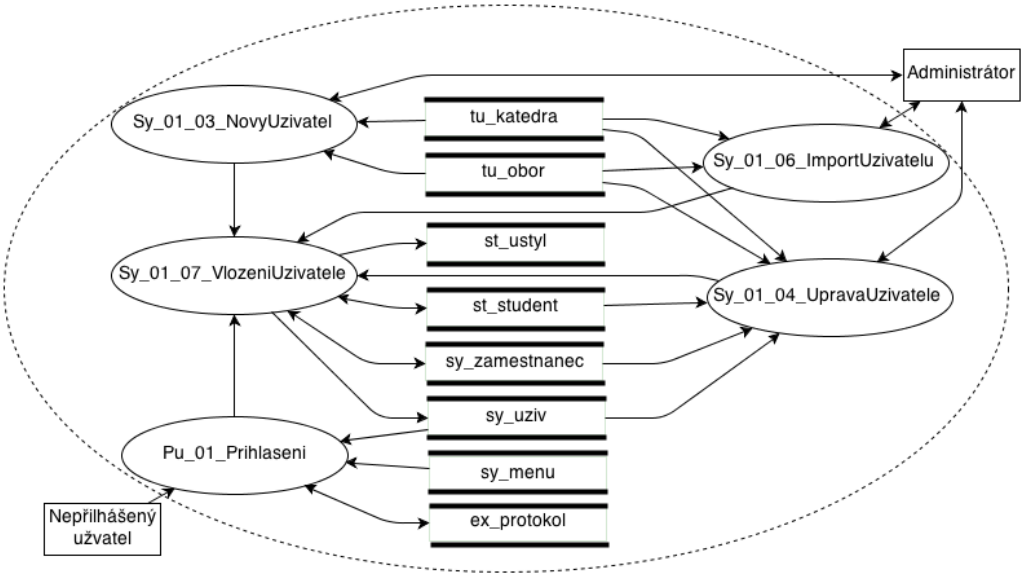


Obrázek 10: DFD 1. úrovně - Student

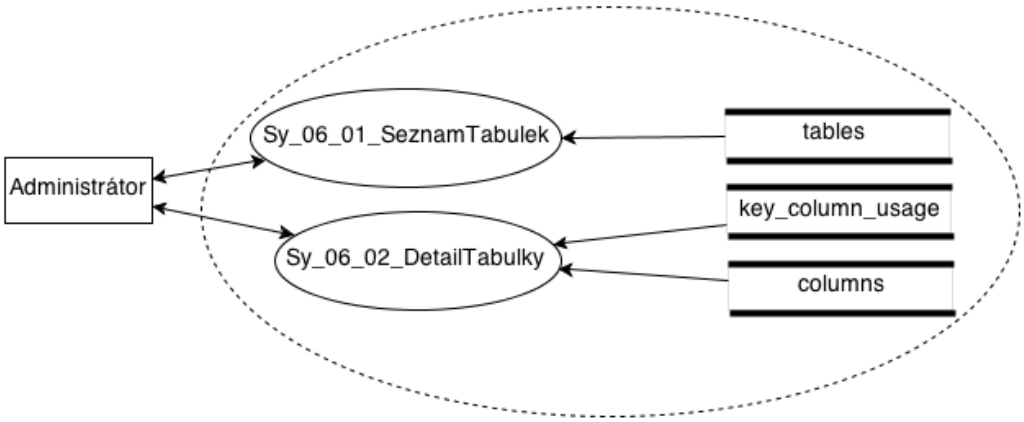


Obrázek 11: DFD 1. úrovně - Systém/administrátor

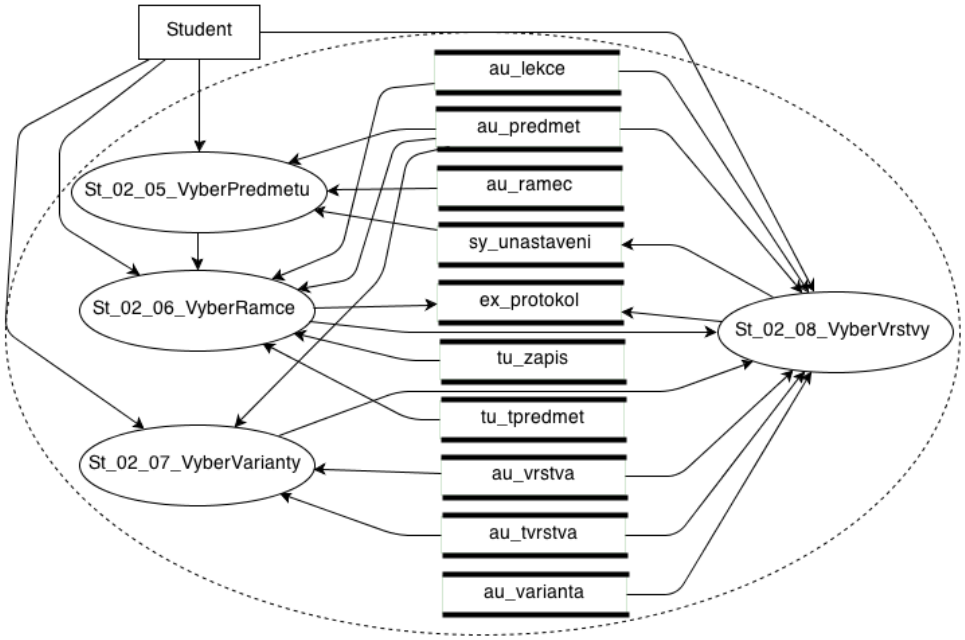
Pro většinu zvolených elementárních funkci, které jsou popsány pomocí případů užití, jsou pro lepší názornost zobrazeny i DFD 2. úrovně (obrázky 12, 13, 14 a 15). Názvy elementárních funkcí se nachází přímo v diagramu a jejich stručný popis z požadavků je uveden v příloze.



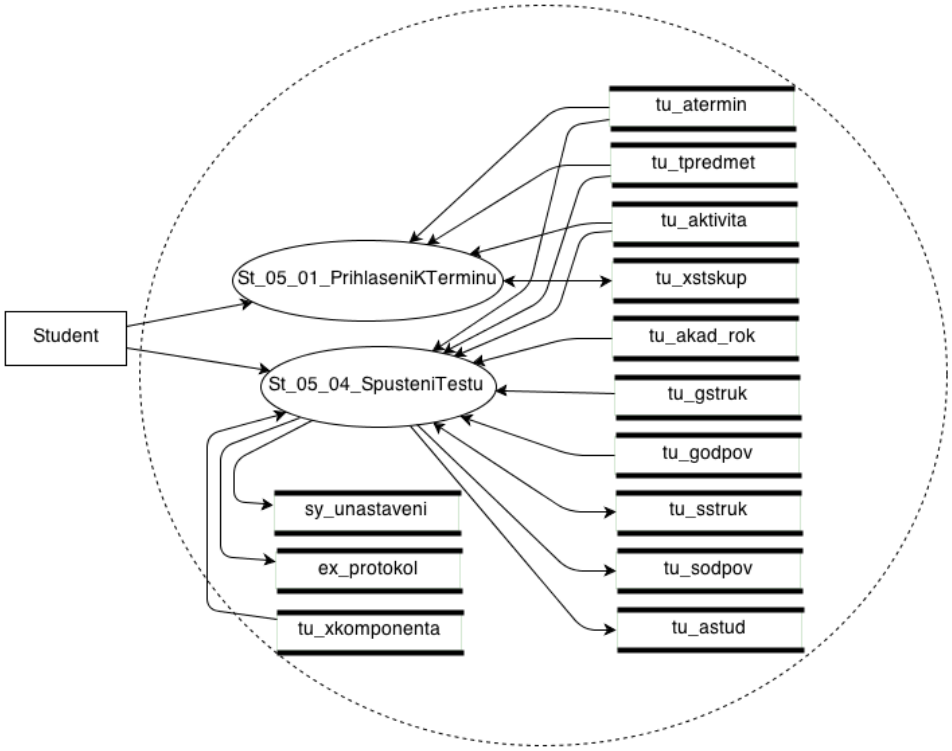
Obrázek 12: DFD 2. úrovně - Uživatelé



Obrázek 13: DFD 2. úrovně - Model databáze



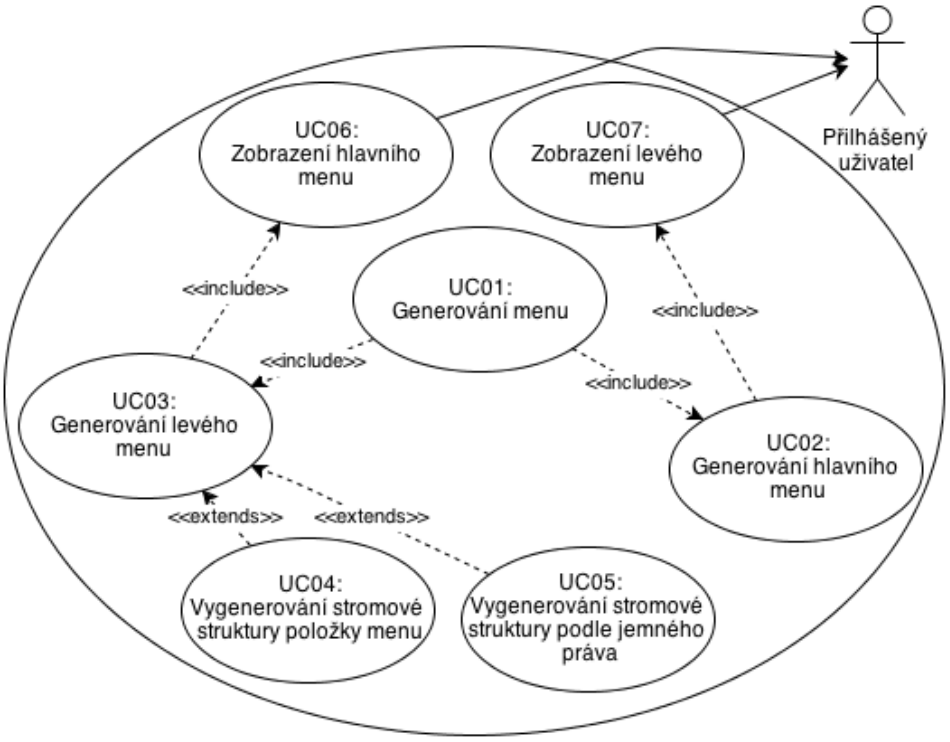
Obrázek 14: DFD 2. úrovně - Výběr ve výuce



Obrázek 15: DFD 2. úrovně - Zlomek z testů

4.2.2.2 Případy užití

V této části funkční analýzy systému jsou uvedeny jednotlivé výše zmíněné případy užití. Ty mimo jiné obsahují i prvky z následující etapy - návrhu systému, jako jsou transakce, databázové příkazy SQL aj.



Obrázek 16: Use case diagram: Generování menu

UC01: Generování menu

Aktér	systém
Úroveň	systémová
Spouštěč	zobrazení stránky obsahující menu
Vstupní podmínky	identifikátor aktuální zvolené položky menu (elementární funkce)

Hlavní scénář

1. Získání čísel rolí *cislaRoli* aktuálního uživatele
    1.1 Výběr jednotlivých rolí (*role_admin*, *role_autor*, *role_expert*, *role_student*, *role_tutor*) pro aktuálního uživatele z tabulky **sy_uziv**
    1.2 Převedení rolí na číselné pole (st = 1, tu = 2, au = 3, ex = 4, ad = 5) *cislaRoli*
2. Získání jemných práv aktuálního uživatele
    2.1 Výběr řetězce s právy (práva k modulům a jemný práva) *prava* z tabulky **sy_uziv**
    2.2 Převedení řetězce *prava* na pole a odstranění práv k modulům *pravaPole*
3. Generování hlavního menu
4. Generování levého menu

UC02: Generování hlavního menu

Aktér	systém
Úroveň	systémová
Spouštěč	UC01
Vstupní podmínky	čísla rolí aktuálního uživatele *cislaRoli*, jemná práva aktuálního uživatele *pravaPole*, identifikátor aktuální zvolené položky menu (elementární funkce)

Hlavní scénář

1. Získání položek menu obsahující moduly
    1.1 Převedení čísel rolí *cislaRoli* na pole s identifikátory modulů *moduly* podle 0. úrovně DFD
    1.2 Získání identifikátorů modulů *modulyPrava* z jemných práv *pravaPole*, ke kterým nemá uživatel přístup (podle role aktuálního uživatele)
    1.3 Získání položek menu na 0. úrovni DFD z tabulky **sy_menu**

```
SELECT id_menu, LOWER(iden_m) AS iden_m, nazev_m, volani_m FROM sy_menu WHERE id_nadm is
    null OR id_nadm = '' AND zobrazit_m = 1 ORDER BY poradi_m ASC
```

    1.4 Pro každý selektovaný řádek se provádí:
        1.4.1 Kontrola, zdali identifikátor *iden_m* patří do *moduly* nebo *modulyPrava*
        1.4.2 Vytvoření nové položky menu reprezentující třída Obsah z databázových údajů do pole *hlavniMenu*
2. Získání položek submenu
    2.1 Výběr položek submenu aktuálního modulu z tabulky **sy_menu**

```
SELECT id_menu, LOWER(iden_m) AS iden_m, nazev_m, volani_m FROM sy_menu WHERE id_nadm = (
    SELECT id_menu FROM sy_menu WHERE (id_nadm is null OR id_nadm = "") AND LOWER(iden_m)
    = LOWER(?)) AND zobrazit_m = 1 ORDER BY poradi_m ASC
```

    2.2 Pokud na aktuální modul má uživatel právo podle některé role z *cislaRoli* (id aktuálního modulu je stejné jako id modulu dané role), pro každý vybraný řádek z kroku 2.1 se vytvoří nová položka třídy Obsah z databázových údajů do pole *subMenu*
    2.3 Výběr položek submenu z jemných práv
        2.3.1 Pro každé jemné právo z pole *prava* se provádí:
            2.3.1.1 Kontrola zdali identifikátor *iden_m* je 1. úrovně DFD
            2.3.1.2 Identifikátor *iden_m* se vloží do pole *jemnaPrava*
        2.3.2 Pro každý vybraný řádek z kroku 2.1 se dále provádí:
            2.3.2.1 Kontrola, zdali má id menu *iden_m* daného řádku stejné první dvě části jako některé id z pole *jemnaPrava*
            2.3.2.2 Odkaz se nastaví stejný jako u položky, ke které má přístup, výběrem *volani_m* z tabulky **sy_menu** podle *iden_m* této položky
            2.3.2.3 Vytvoří se nová položka třídy Obsah z databázových údajů do pole subMenu
3. Zobrazení hlavního menu

Rozšíření
1.4.1a Pokud *iden_m* nepatří, položka se v dalším kroku nevytvoří
1.4.2a Pokud je aktuální modul stejný jako identifikátor položky, položka se označí jako současná
1.4.2b Pokud je položka modulu v menu určena z jemných práv, její odkaz se nastaví stejný jako u položky, ke které má přístup výběrem *volani_m* z tabulky **sy_menu** podle *iden_m* této položky
1.4.2c, 4.2.2a Pokud je odkaz menu *volani_m* prázdný, nastaví se výchozí
Public:Neimplementovano:show
1.4.2d, 4.2.2b Pokud v odkaze menu *volani_m* není vloženo id menu, vloží se *iden_m*
2.2a Pokud jsou první 2 části identifikátoru menu stejné jako u aktuálního id menu, položka se označí jako současná
2.3.1.1a Pokud *iden_m* není 1. úrovně DFD, následující krok se neprovede.
2.3.2.1a Pokud *iden_m* nemá stejné první 2 části, následující 2 kroky se neprovedou
2.3.2.2a Pokud jsou první 2 části aktuálního id menu stejné jako id menu aktuálního řádku, položka se označí jako současná

UC03: Generování levého menu

Aktér	systém
Úroveň	systémová
Spouštěč	UC01
Vstupní podmínky	čísla rolí aktuálního uživatele *cislaRoli*, jemná práva aktuálního uživatele *pravaPole*, identifikátor aktuální zvolené položky menu (elementární funkce)

Hlavní scénář
1. Pokud na aktuální modul má uživatel právo podle některé role z *cislaRoli* (id aktuálního modulu je stejné jako id modulu dané role), provede se Vygenerování stromové struktury položky menu *leveMenu* z aktuální položky submenu
2. Získání titulku levého menu, jedná se o *nazev_m* aktuální nadřazené položky submenu z tabulky **sy_menu**
3. Zobrazení levého menu

Rozšíření
1a Pokud nemá uživatel na aktuální modul právo podle některé role z *cislaRoli*
    1a.1 Generování struktury menu podle jemných práv uživatele
        1a.1.1 Výběr všech jemných práv z *pravaPole*, které mají první 2 části identifikátoru shodné jako aktuální id menu
        1a.1.2 Kontrola, pokud se jedná o právo k položce ze submenu
        1a.1.3 Vygenerování stromové struktury položky menu z aktuální položky submenu do proměnné *leveMenu*
1a.1.2a Pokud se jedná o jemné právo k položkám z nižší úrovně
    1a.1.2a.1 Vygenerování stromové struktury podle jemného práva z každého jemného práva

1a.1.2a.2 Spojení stromových struktur všech jemných práv a vytvoření struktury *leveMenu*

UC04: Vygenerování stromové struktury položky menu

Aktér	systém
Úroveň	systémová
Spouštěč	UC03
Vstupní podmínky	id zpracovávané položky menu, id aktuální zvolené položky menu

Hlavní scénář

1. Výběr podpoložek vstupní zpracovávané položky z tabulky **sy_menu**

```
SELECT id_menu, iden_m, nazev_m, volani_m, zobrazit_m FROM sy_menu WHERE id_nadm = ? ORDER
    BY poradi_m ASC
```

2. Vytvoření pole podpoložek *podpolozky* vstupní položky
   2.1 Kontrola, zdali se položka má zobrazit v menu (*zobrazit_m* má hodnotu true)
   2.2 Vytvoření nové položky levého menu, objekt PolozkaLevehoMenu z databázových dat
   2.3 Vytvořené položce se vygeneruje podřazená stromová struktura Vygenerování stromové struktury položky menu
   2.4 Položka se přidá do pole *podpolozky*

Rozšíření

2.1a Pokud se položka zobrazit v menu nemá, následující 2 kroky se neprovedou
2.2a Pokud je *iden_m* zpracovávané položky stejný jako identifikátor aktuální položky, položka se označí jako současná
2.2b Pokud má zpracovávaná položka aktuální podpoložku menu, která se nezobrazuje v menu, označí se ke zvýraznění
2.2c Pokud je odkaz *volani_m* prázdný, nastaví se výchozí Public:Neimplementovano:show
2.2d Pokud v odkaze *volani_m* není vloženo id menu, vloží se *iden_m*

UC05: Vygenerování stromové struktury podle jemného práva

Aktér	systém
Úroveň	systémová
Spouštěč	UC03
Vstupní podmínky	id zpracovávané položky menu, id aktuální zvolené položky menu, řetězec jemného práva

Hlavní scénář

1. Výběr zpracovávané položky menu z tabulky **sy_menu**

```
SELECT id_menu, LOWER(iden_m) AS iden_m, id_nadm, nazev_m, volani_m FROM sy_menu WHERE
    LOWER(iden_m) = LOWER(?) AND zobrazit_m = 1
```

2. Vytvoření nové položky levého menu *polozka* - objekt PolozkaLevehoMenu z dat
3. Pokud se v jemném právu nevyskytuje akce, generuje se podřazená struktura

Vygenerování stromové struktury položky menu vytvořené položky
4. Vygenerování rodičovské hierarchie z vytvořené položky v cyklu
4.1 Výběr rodičovské položky ze **sy_menu** podle *iden_m* dané položky *polozka*

```
SELECT id_menu, LOWER(iden_m) AS iden_m, id_nadm, nazev_m, volani_m FROM sy_menu WHERE
    id_menu = (SELECT id_nadm FROM sy_menu WHERE LOWER(iden_m) = LOWER(?)) AND
    zobrazit_m = 1
```

4.2 Kontrola, že se nejedná o položku submenu
4.3 Vytvoření rodičovské položky *rodic* - objekt PolozkaLevehoMenu z databázových dat
4.4 Rodičovské položce je nastaven odkaz položky *polozka* z jemného práva
4.5 Vytvořené rodičovské položce je nastavena podpoložka *polozka*
4.6 Položka *polozka* je nahrazena položkou *rodic* a pokračuje se krokem 4.1

Rozšíření
2a, 4.3a Pokud je *iden_m* položky stejný jako identifikátor aktuální položky, položka se označí jako současná
2b Pokud je odkaz *volani_m* prázdný, nastaví se výchozí Public:Neimplementovano:show
2c Pokud v odkaze *volani_m* není vloženo id menu, vloží se *iden_m*
3a Pokud se akce vyskytuje, neprovede se vygenerování podpoložek
4.2a Pokud se jedná o položku submenu, cyklus se zastavuje

UC06: Zobrazení hlavního menu

Aktér	systém
Úroveň	systémová
Spouštěč	UC02
Vstupní podmínky	existují vygenerované objektové položky hlavního menu *hlavniMenu* a submenu *subMenu*

Hlavní scénář
1. Pro každý prvek pole *hlavniMenu* se uživateli zobrazí:
1.2 Označení CSS třídy určující, zdali je položka aktuální
1.3 Vypsání HTML odkazu, jehož název udává hodnota *Obsah->nazev_m* a odkaz *Obsah->volani_m*
2. Pro každý prvek pole *subMenu* se dále uživateli zobrazí:
2.2 Označení css třídy určující, zdali je položka aktuální
2.3 Vypsání html odkazu, jehož název udává hodnota *Obsah->nazev_m* a odkaz *Obsah->volani_m*

UC07: Zobrazení levého menu

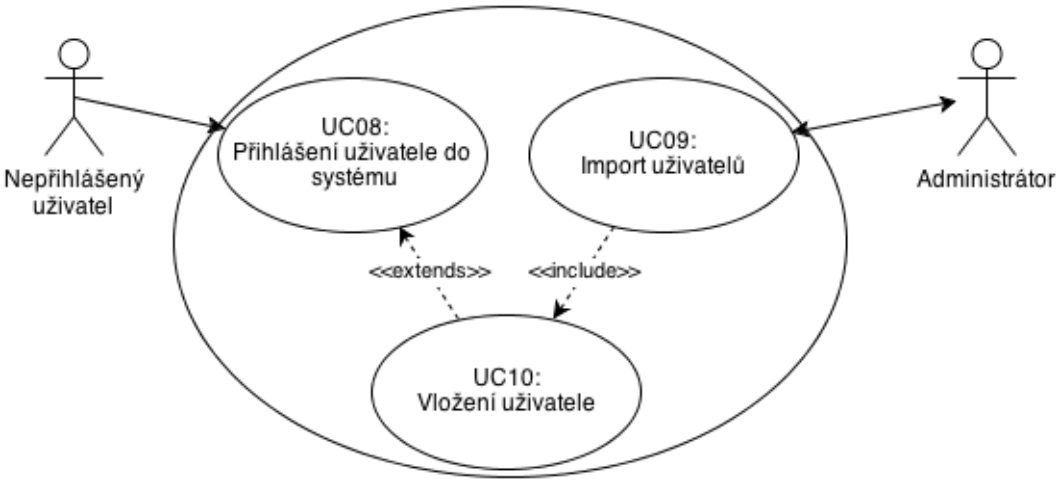
Aktér	systém
Úroveň	uživatelská
Spouštěč	UC03 a elementární funkce používající generátor levého menu
Vstupní podmínky	je vygenerována struktura levého menu *leveMenu*, identifikátor aktuální zvolené položky menu (elementární funkce)

Hlavní scénář

1. Pro každou položku první úrovně levého menu se provádí:
    1.1 Vykreslení položky uživateli - tag
    1.2 Vykreslení HTML tagu <ul> pro začátek nové hierarchie
    1.3 Provedení Zobrazení levého menu pro všechny podpoložky dané položky
    1.4 Vykreslení HTML tagu pro konec hierarchie
2. Zveřejnění aktuální položky menu v zobrazené hierarchické struktuře
    2.1 Všechny prvky levého menu se skryjí až na první úroveň
    2.2 Zobrazení podpoložek jednotlivých úrovní až k aktuální položce

Rozšíření

1.1a Pokud položka je listová a je aktuální, vykreslí se HTML odkaz a zvýrazní se tučně
1.1b Pokud položka je listová a není aktuální, vykreslí se jen HTML odkaz
1.1c Pokud je položka uzlová, je aktuální a nemá odkaz, zvýrazní se tučně a je rozklikávací
1.1d Pokud je položka uzlová, je aktuální a má odkaz, vykreslí se HTML odkaz a zvýrazní se tučně
1.1e Pokud je položka uzlová, není aktuální a nemá odkaz, je rozklikávací
1.1f Pokud je položka uzlová, není aktuální a má odkaz, vykreslí se HTML odkaz



Obrázek 17: Use case diagram: Vložení uživatele

UC08: Přihlášení uživatele do systému (sekvenční diagram se nachází v příloze)

Aktér	nepřihlášený uživatel (student, tutor, autor, expert, administrátor) nebo uživatel, který se vyskytuje ve školní databázi přes LDAP
Úroveň	uživatelská
Spouštěč	uživatel spustí systém

Hlavní scénář
1. Zobrazení formuláře uživateli obsahující formulářové prvky pro *login* a *heslo*
2. Uživatel vyplní přihlašovací *login* a *heslo* a odešle formulář
3. Validace, že *heslo* a *login* nejsou prázdné
4. Kontrola, zdali uživatel s daným *loginem* existuje v databázi v tabulce **sy_uziv**
5. Kontrola, zdali je heslo platné (hash *hesla* je shodný s hashem hesla uloženého v databázi v tabulce **sy_uziv**)
6. Nastavení systému
    6.1 Nastavení expirace přihlášení a session z globální proměnné
    6.2 Získání nového sezení pro protokol a uložení do session z tabulky **ex_protokol**
    6.3 Uložení akce přihlášení do systému do protokolu tabulky **ex_protokol**
    6.4 Inicializace algoritmu pro šifrování identifikátorů
7. Přesměrování na výchozí akci modulu, ke kterému je přiřazena hlavní role uživatele, získanou z tabulky **sy_menu**

Rozšíření
3a Pokud jsou heslo nebo login prázdné, vypíše se upozornění a pokračuje se krokem 1.
4a Pokud uživatel v databázi systému neexistuje
    4a.1 Kontrola, zdali uživatel existuje ve školní databázi přes LDAP
    4a.2 Ověření přes LDAP, že je heslo platné
    4a.3 Vložení nového studenta Vložení uživatele s údaji *login*, *email*, *jména* a *příjmení* z LDAP a pokračuje se krokem 6.
5a, 4a.2a Pokud heslo není platné, vypíše se uživateli hláška o selhání přihlášení a pokračuje se krokem 1.

UC09: Import uživatelů

Aktér	administrátor
Úroveň	uživatelská
Spouštěč	administrátor zvolí položku menu „Import uživatelů“

Hlavní scénář
1. Zobrazení formuláře uživateli obsahující 5 checkboxů pro výběr rolí (*role_student*, *role_tutor*, *role_autor*, *role_expert*, *role_admin*), 5 selectboxů pro výběr hlavní role (představuje atribut *hlavni_role*), 4 selectboxy pro výběr oddělovače dat v souboru (středník, čárka, dvojtečka, svislice) a formulářový prvek pro nahrání souboru
2. Uživatel vyplní všechny údaje formuláře a zvolí soubor s příponou CSV, obsahující seznam uživatelů a odešle formulář
3. Validace vstupních údajů
    3.1 Ověření, že jsou všechny hodnoty vyplněné
    3.2 Ověření, že zvolená hlavní role se nachází ve vybraných rolích
    3.3 Ověření, že se jedná o soubor s příponou .csv
4. Přečtení obsahu souboru a překonvertování do kódování UTF-8
5. Zpracování prvního řádku souboru

5.1 Kontrola, zdali jestli se na prvním řádku nenachází sloupec s chybou
5.2 Do pole s chybami *chyby* se vloží sloupec „chyba“ s příslušným oddělovačem
5.3 Ověření, že první řádek obsahuje povinné sloupce *login*, *jmeno*, *prijmeni* a *email*
6. Zpracování zbylých řádků souboru (pro každý řádek se provádí)
6.1 Kontrola, že aktuální řádek má stejný počet sloupců jako první
6.2 Připraví se asociativní pole *sloupce*, obsahující všechny atributy (sloupce tabulek **sy_uziv** a **st_student**) s prázdnými hodnotami
6.3 Pro každý sloupec z prvního řádku se kontroluje, zdali vůbec existuje ve vygenerovaných sloupcích (klíče pole *sloupce*) a následně do daného sloupce (pole *sloupce*) se zkopíruje hodnota na aktuálním řádku tohoto sloupce
6.4 Pokud není vyplněno *heslo* ani není zvolen atribut *ldap*, *heslo* se nastaví jako *login*
6.5 Zkratky údajů školy (zkratka školy, fakulty, katedry, oboru) se nahradí příslušnými identifikátory z číselníku **tu_obor** a **tu_katedra**, pouze pokud zkratky existují
6.6 Validace vytvořených údajů z daného řádku
6.6.1 Ověření, že povinné hodnoty sloupců *login*, *jmeno*, *prijmeni*, *email* a *ldap* nebo *heslo* jsou vyplněny
6.6.2 Ověření správného tvaru emailu
6.6.3 Ověření, že student s daným loginem v systému již neexistuje
6.6.4 Ověření, že vkládané řetězce nepřekračují povolenou délku
6.7 Vložení uživatele do systému Vložení uživatele s hodnotami ze souboru
7. Vypsání informačního hlášení popisujícího, kolik záznamů bylo vloženo
8. Vypsání chybových hlášení z pole *chyby*, pokud nějaké chyby existují

Rozšíření
3.1a V případě, že nejsou hodnoty vyplněné, vypíše se chybové hlášení *„Vyplňte roli!“*
3.2a V případě, že se hlavní role nenachází ve vybraných, vypíše se chybové hlášení *„Vyplňte hlavní roli příslušnou ke zvolené roli!“*
3.3a V případě, že se nejedná o soubor s příponou CSV, vypíše se chybové hlášení *„Vložte soubor s příponou s .csv!“*
5.1a Pokud se nachází, do pole *chyby* se zkopíruje první řádek souboru
5.3a Pokud soubor neobsahuje alespoň jeden povinný atribut, vypíše se hlášení *„Soubor neobsahuje povinné atributy (login, jmeno, prijmeni, email)!“* a pokračuje se krokem 1.
6.1a Pokud má aktuální řádek jiný počet sloupů, než první, do pole *chyby* se vloží text chyby *„Různý počet sloupců než je v definici!“* popisující s číslem aktuálního řádku. Pokračuje se dalším řádkem souboru
6.6.1a Pokud nějaký povinný atribut vyplněný není, do pole chyb *chyby* se vloží text chyby *„Některé povinné atributy jsou prázdné! Pokud definujete atribut ldap a ne heslo, hodnota musí být 1.“* s číslem řádku a pokračuje se dalším řádkem souboru
6.6.2a Pokud email nemá správný tvar, do pole chyb *chyby* se vloží text chyby *„Email nemá správný tvar!“* s číslem řádku a pokračuje se další řádkem souboru
6.6.3a Pokud uživatel s daným loginem již existuje, do pole *chyby* se vloží text chyby *„Již existuje uživatel s daným loginem!“* s číslem řádku a pokračuje se dalším řádkem souboru
6.6.4a Pokud některý řetězec překračuje maximální délku, do pole *chyby* se vloží text *„Délka některého řetězce je příliš velká!“* s číslem řádku a pokračuje se dalším řádkem

UC10: Vložení uživatele

Aktér	systém
Úroveň	systémová
Spouštěč	UC8, UC9, administrátorská funkce pro vložení uživatele do systému nebo úpravu údajů uživatele
Vstupní podmínky	vkládané hodnoty *hodnoty*

Hlavní scénář

1. Zjištění, zdali se jedná o studenta *jeStudent* podle (hodnota pole role obsahuje číslo 1)
2. Zjištění, zdali se jedná o zaměstnance *jeZamestnanec* podle (hodnota pole role obsahuje jiné číslo než 1)
3. Kontrola, zdali se jedná o vložení nového uživatele (identifikátor *id_uziv* je prázdný)
4. Vložení nového uživatele z *hodnoty* (hodnoty *login*, *heslo*, *ldap*, *prava*, *role_student*, *role_tutor*, *role_autor*, *role_expert*, *role_admin*, *hlavni_role*) do tabulky **sy_uziv**
5. Pokud se jedná o studenta *jeStudent*, vloží se záznam do tabulky **st_student** z *hodnoty* (hodnoty *id_uziv*, *dat_nar*, *jmeno*, *prijmeni*, *titul*, *id_oboru*, *rocnik*, *email*, *virtualni*) a vytvoří se příslušný záznam v tabulce **st_ustyl** s identifikátorem *id_uziv* a zbylými prázdnými hodnotami
6. Pokud se jedná o zaměstnance *jeZamestnanec*, vloží se z pole hodnoty záznam do tabulky **sy_zamestnanec** (hodnoty *id_uziv*, *dat_nar*, *jmeno*, *prijmeni*, *titul*, *email*, *id_katedry*, *mistnost*, *telefon*)

Rozšíření

3a Pokud se jedná o editaci záznamu (identifikátor *id_uziv* není prázdný)

- 3a.1 Ověření, zdali editovaný uživatel již v tabulce **st_student** existuje a je studentem, a hodnota se vloží do proměnné *existujeStudent*
- 3a.2 Z tabulky **sy_zamestnanec** se ověří, zdali editovaný uživatel již je zaměstnanec, a hodnota se vloží do proměnné *existujeZamestnanec*
- 3a.3 Změní se upravené údaje v tabulce **sy_uziv**
- 3a.4 Pokud se jedná o studenta *jeStudent* a student v databázi existuje *existujeStudent*, změní se upravené hodnoty v tabulce **st_student**
- 3a.5 Pokud se jedná o studenta *jeStudent* a student v databázi neexistuje (neplatí *existujeStudent*), provede se krok 5.
- 3a.6 Pokud se nejedná o studenta (neplatí *jeStudent*) a student v databázi existuje *existujeStudent*, student s daným *id_uziv* se smaže z tabulky **st_student** a **st_ustyl**
- 3a.7 Pokud se jedná o zaměstnance *jeZamestnanec* a zaměstnanec v databázi existuje *existujeZamestnanec*, změní se upravené hodnoty v tabulce **sy_zamestnanec**
- 3a.7 Pokud se jedná o zaměstnance *jeZamestnanec* a zaměstnanec v databázi neexistuje (neplatí *existujeZamestnanec*), provede se krok 6.
- 3a.8 Pokud se nejedná o zaměstnance (neplatí *jeZamestnanec*) a v databázi existuje *existujeZamestnanec*, zaměstnanec s daným *id_uziv* se smaže z tabulky **sy_zamestnanec**

UC11: Zobrazení seznamu databázových tabulek

Aktér	administrátor
Úroveň	uživatelská
Spouštěč	administrátor zvolí položku menu „Model databáze“

Hlavní scénář

1. Z databázového katalogu se vyberou všechny tabulky

```
SELECT TABLE_NAME, TABLE_COMMENT, TABLE_ROWS FROM tables WHERE TABLE_SCHEMA = '
    barborka4'
```

2. Pro každou tabulku se z názvu (prefix modulu) určí, ke kterému modulu patří a přidá se sloupec s názvem modulu
3. Uživateli se zobrazí tabulka obsahující všechny databázové tabulky

UC12: Zobrazení detailu databázové tabulky

Aktér	administrátor
Úroveň	uživatelská
Spouštěč	administrátor vybere tabulku, pro kterou chce zobrazit detail

Hlavní scénář

1. Výběr a zobrazení všech atributů zvolené databázové tabulky

```
SELECT COLUMN_NAME, COLUMN_COMMENT, COLUMN_TYPE, IS_NULLABLE FROM COLUMNS
    WHERE TABLE_NAME = ? AND TABLE_SCHEMA = 'barborka4'
```

2. Pro každý atribut se zobrazí:
   2.1 Označení, zdali je atribut primární klíč (pokud existuje klíč dané tabulky v katalogu)

```
SELECT COUNT(*) FROM KEY_COLUMN_USAGE WHERE TABLE_NAME = ? AND COLUMN_NAME = ?
    AND CONSTRAINT_NAME = 'PRIMARY' AND TABLE_SCHEMA = 'barborka4'
```

   2.2 Označení, zdali je atribut cizí klíč a výběr názvu tabulky referenční tabulky z katalogu

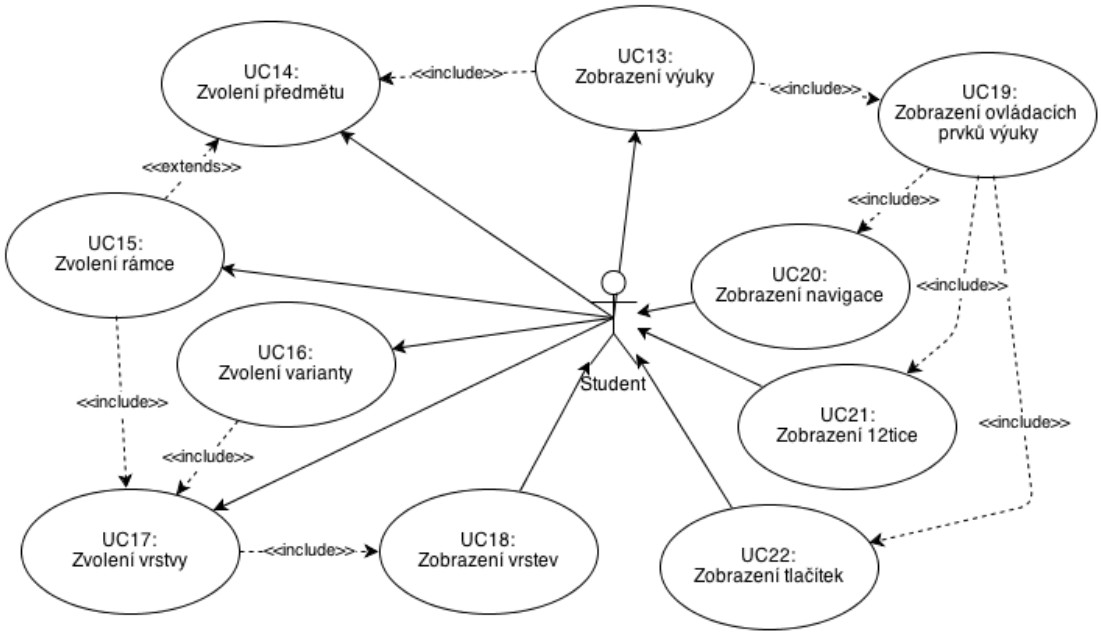
```
SELECT REFERENCED_TABLE_NAME FROM KEY_COLUMN_USAGE WHERE TABLE_NAME = ? AND
    COLUMN_NAME = ? AND REFERENCED_TABLE_NAME is not NULL AND TABLE_SCHEMA = '
    barborka4'
```

3. Zobrazení lineárního zápisu tabulky výběrem atributů z katalogu

```
SELECT COLUMN_NAME FROM COLUMNS WHERE TABLE_NAME = ? AND TABLE_SCHEMA = '
    barborka4'
```

4. Zobrazení seznamu závislých tabulek na zvolené tabulce výběrem z katalogu

```
SELECT u.table_name FROM KEY_COLUMN_USAGE AS u WHERE u.table_schema='barborka4' AND u.
    referenced_table_name = ?
```



Obrázek 18: Use case diagram: Zobrazení výuky

UC13: Zobrazení výuky

Aktér	student
Úroveň	uživatelská
Spouštěč	student zvolí položku submenu „Výuka“ pro spuštění výuky

Hlavní scénář

1. Kontrola, zdali není spuštěn jiný test či autotest (ze session)
2. Zvolení předmětu
3. Zobrazení ovládacích prvků výuky

Rozšíření

1a Pokud je spuštěn test či autotest, zobrazí se informace o nedostupnosti výuky a výuka se nezobrazí

UC14: Zvolení předmětu

Aktér	systém, student
Úroveň	systémová
Spouštěč	UC13, student zvolí předmět v navigaci

Hlavní scénář

1. Kontrola, zdali existuje globální filtr pro autorský předmět a nastaví se z filtru
2. Kontrola, zdali v session existuje poslední otevřený rámec ze studentova osobního nastavení nebo zdali je vybrán předmět
3. Pokud v session existuje poslední zobrazený rámec *id_ramec*, ale neexistuje předmět

3.1 Výběr předmětu *au_predmet* podle rámce *id_ramec* z tabulky **au_predmet**
3.2 Načtení osobního uživatelského nastavení uživatele pro předmět *au_predmet* z databázové tabulky **sy_unastaveni** do session
3.3 Zvolení rámce z identifikátoru *id_ramec*

Rozšíření
1a Pokud se jedná o filtr s tutorským předmětem, vybere se identifikátor příslušného autorského *id_pred* z databázové tabulky **au_predmet**
2a Pokud podmínka neplatí, není výuka zobrazena, use case je ukončen a zobrazí se pouze navigace s předměty
3a Pokud zvolený předmět (*id_pred*) existuje
3a.1 Načtení osobního uživatelského nastavení uživatele pro předmět *au_predmet* z databázové tabulky **sy_unastaveni** do session
3a.2 Kontrola session, zdali neobsahuje poslední zobrazený rámec daného předmětu
3a.3 Výběr prvního rámce *id_ramec* zvoleného předmětu *id_pred* z tabulky **au_ramec**
3a.4 Zvolení rámce z identifikátoru *id_ramec*
3a.2a Pokud session obsahuje poslední zobrazený rámec předmětu, provede se Zvolení rámce

UC15: Zvolení rámce

Aktér	systém, student
Úroveň	systémová
Spouštěč	UC13, student zvolí rámec v navigaci
Vstupní podmínky	identifikátor rámce *id_ramec*, který je zvolen

Hlavní scénář
1. Kontrola, zdali má student oprávnění k zobrazení daného rámce z tabulky (rámec se nachází v předmětu, který má student zapsaný **tu_zapis** nebo předmět je veřejný z tabulky **tu_tpredmet**)
2. Výběr aktuálního výukového stylu daného studenta virtuálním učitelem a získání doporučených vrstev *doporucene*
3. Pokud není zvolen předmět, předmět *id_pred* se nastaví podle rámce výběrem z tabulky **au_predmet**
4. Aktualizace session a protokolu
4.1 Pokud zvolený rámec *id_ramec* je různý od aktuálního rámce (v session), vybere se z *id_ramec* lekce, ve které se nachází (*id_lekce* z tabulky **au_lekce**)
4.2 Kontrola, pokud je lekce *id_lekce* různá od aktuální lekce (v session) a předmět *id_pred* různý od aktuálního předmětu
4.3 Do session se nastaví nový aktuální předmět *id_pred* a uloží se záznam o přihlášení k předmětu do protokolu (tabulka **ex_protokol**)
4.4 Do session se nastaví nová aktuální lekce *id_lekce* a uloží se záznam o přihlášení k lekci do protokolu (tabulka **ex_protokol**)
4.5 Do session se nastaví nový aktuální rámec *id_ramec* a uloží se záznam o přihlášení k rámci do protokolu (tabulka **ex_protokol**)

5. Kontrola, zdali nedošlo ke změně výukového stylu uprostřed výuky rámce porovnáním vrstev *doporucene* a aktuální doporučených v session
6. Zvolení vrstvy (aktuální vrstva rámce ze session)

Rozšíření
1a Nemá-li student oprávnění k rámci nebo řetězec *id_ramec* je prázdný, výuku nelze zobrazit
4.1a Pokud jsou identifikátory rámců stejné, pokračuje se krokem 5.
4.2a Pokud jsou identifikátory lekcí stejné, pokračuje se krokem 4.5
4.3a Pokud jsou identifikátory předmětů stejné, pokračuje se krokem 4.4
5a Došlo-li ke změně, rámec se resetuje a zobrazí se první vrstva z *doporucene*, aktualizují se zobrazené vrstvy, vrstvy k zobrazení (nastaví se přímo všechny doporučené *doporucene*) a aktuální zobrazená v session

UC16: Zvolení varianty

Aktér	student
Úroveň	systémová
Spouštěč	student zvolí variantu v navigaci 12tice
Vstupní podmínky	identifikátor zvolené vrstvy dané varianty (*id_vrstva* či *id_tvrstva*)

Hlavní scénář
1. Pokud není zvolen předmět, získá se výběrem z tabulky **au_predmet** podle vrstvy *id_vrstva* či *id_tvrstva*
2. Kontrola, zdali aktuální vrstva a zvolená vrstva (*id_vrstva* či *id_tvrstva*) patří ke stejnému rámci a zdali jsou stejného typu (v ukázce dotazu kontroly pro testovací vrstvu je na vstupu *id_tvrstva*, *id_tvrstva* a aktuální vrstva) výběrem z tabulky **au_vrstva** či **au_tvrstva**

```
SELECT COUNT(*) FROM au_tvrstva AS v
   INNER JOIN au_varianta AS a ON v.id_vari = a.id_vari
   WHERE v.id_cvr = (SELECT id_cvr
   FROM au_tvrstva WHERE id_tvrstva = ?)
      AND a.id_ramec = (SELECT a.id_ramec
      FROM au_varianta AS a
         INNER JOIN au_tvrstva AS v ON a.id_vari = v.id_vari
      WHERE v.id_tvrstva = ?)
   AND v.id_tvrstva = ?
```

3. Virtuálním učitelem se provede změna vrstvy a aktualizuje se seznam zobrazených vrstev, seznam vrstev k zobrazení a aktuální vrstva rámce (*id_vrstva* či *id_tvrstva*) v session
4. Zvolení vrstvy (aktuální vrstva rámce ze session)

Rozšíření
2a Pokud podmínka neplatí, výuku nelze zobrazit

UC17: Zvolení vrstvy

Aktér	systém, student
Úroveň	systémová
Spouštěč	UC15, UC16, student zvolí zobrazenou vrstvu
Vstupní podmínky	identifikátor vrstvy *id_vrstva* či *id_tvrstva*, která je zvolena

Hlavní scénář

1. Kontrola, zdali existuje aktuální rámec (v session)
2. Kontrola, zdali zvolená vrstva se nachází v zobrazených vrstvách rámce (v session)
3. Pokud neexistuje aktuální předmět, zkontroluje se, zdali existuje globální filtr pro autorský předmět a nastaví se z filtru
4. Pokud je aktuální předmět prázdný, vybere se podle zvolené vrstvy z tabulky **au_predmet**
5. Uložení uživatelského nastavení (poslední zobrazený rámec v rámci všech předmětů, poslední zobrazený rámec v rámci aktuálního předmětu, vrstvy k zobrazení, doporučené vrstvy, zobrazené vrstvy a aktuální označené vrstvy rámců předmětu) ze session do tabulky **sy_unastaveni**
6. Aktualizace session a protokolu
    - 6.1 Pokud je zvolená vrstva *id_vrstva* různá od aktuální (v session), vybere se varianta *id_vari* z vrstvy (tabulka **au_vrstva** či **au_tvrstva**)
    - 6.2 Pokud je varianta *id_vari* různá jako aktuální (v session), vybere se rámec *id_ramec* z přiřazené varianty *id_vari* (tabulka **au_varianta**)
    - 6.3 Pokud je zvolený rámec *id_ramec* různý od aktuálního (v session), vybere se *id_ramec* z lekce *id_lekce*, ve které se nachází, z tabulky **au_lekce**
    - 6.4 Kontrola, pokud je lekce *id_lekce* různá od aktuální lekce (v session) a předmět *id_pred* různý od aktuálního předmětu
    - 6.5 Do session se nastaví nový aktuální předmět *id_pred* a uloží se záznam o přihlášení k předmětu do protokolu (tabulka **ex_protokol**)
    - 6.6 Do session se nastaví nová aktuální lekce *id_lekce* a uloží se záznam o přihlášení k lekci do protokolu (tabulka **ex_protokol**)
    - 6.7 Do session se nastaví nový aktuální rámec *id_ramec* a uloží se záznam o přihlášení k rámci do protokolu (tabulka **ex_protokol**)
    - 6.8 Do session se nastaví nová aktuální varianta *id_vari* a uloží se záznam o přihlášení k variantě do protokolu (tabulka **ex_protokol**)
    - 6.9 Do session se nastaví nová aktuální vrstva *id_vrstva* či *id_tvrstva* a uloží se záznam o přihlášení k vrstvě či testovací vrstvě do protokolu (tabulka **ex_protokol**)
7. Zobrazení vrstev

Rozšíření

1a Pokud aktuální rámec neexistuje, vybere se z databáze podle vložené vrstvy z tabulky **au_varianta** a provede se redirect na zobrazení tohoto rámce

2a Pokud má identifikátor prázdný řetězec nebo se vrstva nenachází v zobrazených vrstvách rámce, nelze výuka zobrazit

3a Pokud se jedná o filtr s tutorským předmětem, vybere se identifikátor příslušného autorského *id_predmet* z databázové tabulky **au_predmet**

6.1a Pokud jsou identifikátory vrstev stejné, use case končí
6.2a Pokud jsou identifikátory variant stejné, pokračuje se krokem 6.9
6.3a Pokud jsou identifikátory rámců stejné, pokračuje se krokem 6.8
6.4a Pokud jsou identifikátory lekcí stejné, pokračuje se krokem 6.7
6.5a Pokud jsou identifikátory předmětů stejné, pokračuje se krokem 6.6

UC18: Zobrazení vrstev

Aktér	systém
Úroveň	uživatelská
Spouštěč	UC17

Hlavní scénář

1. Generování metadat vrstev z identifikátorů označených v session jako zobrazené vrstvy (pro každou vrstvu):
    1.1 Určení, zdali se jedná o aktuální vrstvu, změněnou vrstvu (jiná než doporučená), vrstvu, která se má zobrazit, o testovací vrstvu a určení hlavní varianty
    1.2 Pro testovací vrstvu se vyberou z tabulky **au_tvrstva** atributy *typ_o*, *nazev_tvr*, *pocet_odpovedi*, *max_bodu*
    1.3 Generování metadat pro komponenty, které jsou obsaženy v dané vrstvě
        1.3.1 Získání metadat z tabulky **au_kompon** atributy *obsah_k*, *zobraz_k*, *multibutton_k*, *typ_ck*, *id_soub*, *nazev*, *cesta*, *mime* (ukázka dotazu pouze pro netestovací vrstvu)

```
SELECT k.obsah_k, k.zobraz_k, k.multibutton_k, c.typ_ck, s.id_soub, s.nazev, s.cesta, s.mime FROM
    au_kompon AS k
  INNER JOIN au_xvrs AS x ON x.id_komp = k.id_komp
  INNER JOIN au_ckomp AS c ON k.id_ckomp = c.id_ckomp
  LEFT JOIN sy_soubor AS s ON k.id_soubor_k = s.id_soub
  WHERE x.id_vrstva = ? ORDER BY x.poradi_k_r ASC
```

        1.3.2 Vygenerování metadat souborů *id*, *ticket*, *hash* využitím služby pro soubory, pracující s tabulkou **sy_soubor**
2. Generování obsahu pro doplňovačky *doplnovacky* – pro každou testovací vrstvu typu doplňovačka se generuje speciální struktura, obsahující pole složené z textových částí a částí s vlastními identifikátory určujícími typy formulářových prvků
3. Dynamická tvorba formulářů
    3.1 Vytvoření formulářů variantních otázek, pro každou variantní otázku (vrstvu) se provádí:
        3.1.1 Kontrola, pokud je nastavení vrstvy uloženo v databázi
        3.1.2 Výběr hodnot nastavení variantních otázek z databáze z tabulky **st_ntvrstva**
        3.1.3 Podle stavu *stav* každé otázky se vytvoří formulářové prvky
    3.2 Vytvoření formulářů tvořených otázek, pro každou tvořenou otázku (vrstvu) se provádí:
        3.2.1 Výběr hodnot nastavení tvořených otázek z databáze z tabulky **st_ntvrstva**
        3.2.2 Podle stavu *stav* každé otázky se vytvoří formulářové prvky
    3.3 Vytvoření formulářů tvořených otázek typu doplňovačka

3.3.1 Výběr nastavení testovací vrstvy z tabulky **st_ntvrstva**
3.3.2 Pro každou část doplňovačky z pole *doplnovacky* se vytváří daný formulářový prvek
3.3.3 Podle stavu *stav* každé otázky se doplní formulářové prvky

4. Zobrazení vrstev studentovi a pro každou vrstvu se provádí:
4.1 Zobrazení pozadí vrstvy, pokud je změněná varianta
4.2 Zobrazení tlačítka pro výběr vrstvy uživatelem
4.3 Zobrazení multibuttonu, pokud se má zobrazit
4.4 Zobrazení všech komponent vrstvy dle vytvořených metadat pod sebou
4.5 Pokud se jedná o testovací vrstvu, vypíší se formulářové prvky pro odpovědi podle vytvořených metadat

Rozšíření

3.1.2a V případě, že nastavení není uloženo, generují se nové odpovědi
3.1.2a.1 Výběr metadat testovací vrstvy *typ_o*, *nazev_tvr*, *pocet_odpovedi*, *max_bodu* z tabulky **au_tvrstva**
3.1.2a.2 Výběr náhodných variantních odpovědí podle typu (1 nebo více správných odpovědí)
3.1.2a.3 Pokud je *typ_o* délky 3, odpovědi se zamíchají
3.1.2a.4 Uložení výchozího nastavení variantních otázek – atributů *id_uziv*, *id_tvrstva*, *rezim*, *stav*, *odpoved*, *zprava* do tabulky **st_ntvrstva**
3.1.3a Pokud se jedná o stav označený 0, vytvoří se prázdný seznam checkboxů s vybranými odpověďmi
3.1.3b Pokud se jedná o stav označený 1, vytvoří se seznam checkboxů s odpověďmi, studentovým vyplněním a obrázkem značícím chybu
3.1.3c Pokud se jedná o stav označený 2, vytvoří se seznam checkboxů s odpověďmi, studentovým vyplněním a obrázkem značícím správné odpovědi
3.1.3d Pokud se jedná o stav označený 3, vytvoří se seznam checkboxů s označením správných odpovědí a obrázkem značícím správné odpovědi
3.1.3e Pokud se jedná o stav označený 4, vytvoří se seznam checkboxů s označením správných odpovědí, vysvětlením špatných odpovědí a obrázkem značícím správné odpovědi
3.1.3f Pokud existuje expertova reakce, tak se vypíše jako chybová hláška nad formulář
3.2.2a Pokud stav neexistuje, vytvoří se prázdný formulářový input
3.2.2b Pokud je stav označen 1, vytvoří se input s vyplněnou odpovědí uživatele *odpoved* a obrázkem značícím chybu
3.2.2c Pokud je stav označen 2, vytvoří se input s vyplněnou odpovědí uživatele *odpoved* a obrázkem značícím správnou odpověď
3.2.2d Pokud existuje expertova reakce, tak se vypíše jako chybová hláška nad formulář
3.3.2a Pokud se jedná o 1 správnou odpověď, vloží se formulářový input box
3.3.2b Pokud se jedná o více správných odpovědí, vloží se formulářový select box
3.3.3a Pokud stav neexistuje, formulářové prvky s doplňovačkou jsou prázdné
3.3.3b Pokud je stav označen 1, formulářové prvky doplňovačky se nastaví studentem vyplněnými hodnotami a vytvoří se obrázek značící chybnou odpověď

3.3.3c Pokud je stav označen 2, formulářové prvky doplňovačky se nastaví studentem vyplněnými hodnotami a vytvoří se obrázek značící správné odpovědi
3.3.3d Pokud existuje expertova reakce, tak se vypíše jako chybová hláška nad formulář
4.5a Pokud se jedná o testovací vrstvu typu doplňovačka, prochází se každá část a pokud je část textová, vypíše text, jinak se zobrazí daný formulářový prvek (input nebo select box)

UC19: Zobrazení ovládacích prvků výuky

Aktér systém
Úroveň uživatelská
Spouštěč UC13
Hlavní scénář
1. Zobrazení navigace
2. Zobrazení 12tice
3. Zobrazení tlačítek

UC20: Zobrazení navigace

Aktér systém
Úroveň uživatelská
Spouštěč UC19
Hlavní scénář
1. Kontrola, zdali předmět není vybrán, tedy se nenachází v globálním filtru
2. Výběr všech předmětů studenta (zapsané, veřejné) z databázové tabulky **au_predmet**

```
SELECT a.id_pred, a.nazev_p FROM au_predmet AS a
  INNER JOIN tu_tpredmet AS p ON p.id_pred_au = a.id_pred
  WHERE p.id_tpred IN (SELECT id_tpred FROM tu_zapis WHERE id_uzivatel = ?) OR p.prihlasovani = 'N
      ' ORDER BY a.nazev_p ASC
```

3. Pro každý předmět se provádí generování struktury levého menu *menu*
    3.1 Vytvoření položky menu třídy PolozkaLevehoMenu *polozka* zastupující daný předmět
    3.2 Výběr všech lekcí daného předmětu z tabulky **au_lekce**
    3.3 Pro každou lekci se provádí:
        3.3.3 Vytvoření položky menu třídy PolozkaLevehoMenu zastupující danou lekci
        3.3.4 Výběr všech rámců dané lekce z tabulky **au_ramec**
        3.3.5 Vytvoření položky menu třídy PolozkaLevehoMenu pro každý rámec dané lekce
        3.3.6 Nastavení rámců jako podpoložky dané lekce
    3.4 Nastavení lekcí jako podpoložky daného předmětu
    3.5 Do *menu* se vloží polozka *polozka*
4. Zobrazení levého menu ze struktury *menu*

Rozšíření
1. Pokud předmět vybrán je, pokračuje se krokem 3.1 pro vybraný předmět

3.1a Odkaz položky menu se nastaví na výběr předmětu
3.3.3a Není nastaven žádný odkaz položky menu, jelikož se jedná o rozklikávací položku
3.3.5a Odkaz položky menu se nastaví na zobrazení daného rámce
3.3.5b Pokud je id rámce *id_ramec* stejný jako identifikátor aktuálního rámce, položka se označí jako aktuální

UC21: Zobrazení 12tice

Aktér	systém
Úroveň	uživatelská
Spouštěč	UC19
Vstupní podmínky	identifikátor vrstvy *id_vrstva* či *id_tvrstva*, která je zvolena

Hlavní scénář
1. Výběr všech vrstev pro 12tici podle aktuální zobrazené vrstvy – všechny vrstvy, které mají varianty z daného rámce, stejný typ jako má *id_vrstva* či *id_tvrstva* a první poradí v multivrstvě (v ukázce dotazu pro testovací je vrstvu na vstupu *id_tvrstva*, *id_tvrstva*)

```
SELECT v.id_tvrstva, a.hloub_v, a.vver, a.vviz, a.vaud, a.vkin
   FROM au_tvrstva AS v
   INNER JOIN au_varianta AS a ON v.id_vari = a.id_vari
   INNER JOIN (SELECT vr.id_vari, vr.id_cvr, MIN(vr.poradi_tvr) AS poradi
      FROM au_tvrstva AS vr
      GROUP BY vr.id_vari, vr.id_cvr) AS x
   ON v.id_vari = x.id_vari AND v.poradi_tvr = x.poradi AND v.id_cvr = x.id_cvr
   WHERE v.id_cvr = (SELECT id_cvr
      FROM au_tvrstva
      WHERE id_tvrstva = ?)
   AND a.id_ramec = (SELECT a.id_ramec
      FROM au_varianta AS a
      INNER JOIN au_tvrstva AS v ON a.id_vari = v.id_vari
      WHERE v.id_tvrstva = ?)
```

2. Odstranění nedostupných variant vrstev - logika virtuálního učitele
3. Pro každou variantu vrstvy se provede:
   3.1 Určení, zdali se jedná o aktuální či doporučenou vrstvu
   3.2 Výběr hlavní smyslové varianty vrstvy *hlavni*
   3.3 Zobrazení varianty na dané pozici v 12tici (podle hlavní varianty *hlavni* a *hloub_v*) jako obarvený čtverec zobrazený v kapitole 2.4.4.1

Rozšíření
3.3a Podle typu hlavní varianty se vykreslí daná barva na pozadí čtverce dané vrstvy
3.3b Pokud se jedná o doporučenou vrstvu, zobrazí se čárkovaný rámeček kolem vrstvy
3.3c Pokud se jedná o aktuální vrstvu, zobrazí se černá tečka uprostřed čtverce

UC22: Zobrazení tlačítek

Aktér	systém
Úroveň	uživatelská
Spouštěč	UC19

Hlavní scénář

1. Výběr a zobrazení odkazu pro předchozí rámec (zobrazeny 2 dotazy jdoucí za sebou, z nichž je vybrán první vrácený *id_ramec*) z tabulky **au_ramec**

```
SELECT id_ramec FROM au_ramec WHERE poradi_r = (SELECT poradi_r − 1 FROM au_ramec WHERE
    id_ramec = ?) AND id_lekce = (SELECT id_lekce FROM au_ramec WHERE id_ramec = ?)
```

```
SELECT id_ramec FROM au_ramec WHERE id_lekce =
  (SELECT id_lekce FROM au_lekce
    WHERE poradi_l =
      (SELECT l.poradi_l − 1 FROM au_lekce AS l
        INNER JOIN au_ramec AS r ON l.id_lekce = r.id_lekce
        WHERE r.id_ramec = ?)
      AND id_pred =
      (SELECT l.id_pred FROM au_lekce AS l
        INNER JOIN au_ramec AS r ON l.id_lekce = r.id_lekce
        WHERE r.id_ramec = ?))
  ORDER BY poradi_r DESC LIMIT 1
```

2. Výběr a zobrazení odkazu pro další rámec (analogicky jako u kroku 1., ale pořadí se přičítá)
3. Pokud je počet aktuálních zobrazených vrstev menší než počet vrstev k zobrazení v session, dojde k zobrazení živého tlačítka pro pokračování ve výuce
4. Zobrazení tlačítka pro reset aktuálního rámce

Rozšíření

3a Pokud je stejný počet vrstev zobrazených vrstev a vrstev k zobrazení v session, z tabulky **au_ramec** se získá další dostupný rámec *id_ramec*, ve kterém když je prázdný, tlačítko se nastaví jako neživé

UC23: Přihlášení k testu (sekvenční diagram se nachází v příloze)

Aktér	systém
Úroveň	uživatelská
Spouštěč	student zvolí položku menu „Přihlášení k testu“

Hlavní scénář

1. Výběr dostupných termínů zapsaných předmětů aktuálního uživatele v aktuálním akademickém roce (typ aktivity je test, termín je dostupný přes přihlášení, aktuální čas přihlášení testu se nachází v daném časovém intervalu, student má předmět zapsaný v aktuálním akademickém roce a student již není k jinému termínu testu zapsaný) z tabulky **tu_atermin** a zbylých uvedených v dotazu (vstupem jsou atributy *id_akad_rok*, *id_akad_rok*, *id_uziv*, *id_uziv*):

```
SELECT t.id_atermin, p.nazev AS predmet, COUNT(x.id_stskup) AS prihlaseni, CONCAT(a.dobaA, '␣min')
    AS cas, t.mistnost, t.kapacita, DATE_FORMAT(t.zacatek,'%H:%i␣%d.%m.%Y') AS zacatek, a.nazev, t
    .nazev_terminu, DATE_FORMAT(t.prihlasit_od,'%d.%m.%Y␣%H:%i') AS prihlasit_od, DATE_FORMAT(t
    .prihlasit_do,'%d.%m.%Y␣%H:%i') AS prihlasit_do, Prihlasit AS akce FROM tu_atermin AS t
  INNER JOIN tu_aktivita AS a ON t.id_aktivity = a. id_aktivity
  INNER JOIN tu_tpredmet AS p ON a.id_tpred = p.id_tpred
  INNER JOIN tu_zapis AS z ON a.id_tpred = z.id_tpred
  LEFT JOIN tu_xstskup AS x ON t.id_skupina = x.id_skupina
  WHERE a.id_typAktivity = 'TypA000002' AND t.prihlasit_od < NOW() AND t.prihlasit_do > NOW() AND
      a.id_akad_rok = ? AND z.id_akad_rok = ? AND z.id_uzivatel = ? AND t.prihlas = 1 AND t.konec >
      NOW() AND a.id_aktivity NOT IN (SELECT ak.id_aktivity FROM tu_aktivita AS ak
    INNER JOIN tu_atermin AS at ON ak.id_aktivity = at. id_aktivity
    INNER JOIN tu_xstskup AS xs ON at.id_skupina = xs.id_skupina
    WHERE at.konec > NOW() AND xs.id_student = ?)
  GROUP BY t.id_atermin
  HAVING COUNT(x.id_stskup) < t.kapacita
```

2. Zobrazení tabulky obsahující seznam termínů se sloupci: *nazev*, *predmet*, *mistnost*, *zacatek*, *cas*, *prihlaseni*, *kapacita*, *prihlasit_od*, *prihlasit_do*, *akce* (odkaz pro přihlášení)
3. Student se přihlásí na předmět zvolením odkazu „Přihlásit“ u vybraného termínu
4. Kontrola, zdali termín je studentovi ještě přístupný z tabulky **tu_atermin** se vstupními atributy *id_akad_rok*, *id_akad_rok*, *id_uziv*, *id_atermin*, *id_uziv*

```
SELECT COUNT(*) FROM tu_atermin AS t
  INNER JOIN tu_aktivita AS a ON t.id_aktivity = a. id_aktivity
  INNER JOIN tu_tpredmet AS p ON a.id_tpred = p.id_tpred
  INNER JOIN tu_zapis AS z ON a.id_tpred = z.id_tpred
  WHERE a.id_typAktivity = 'TypA000002' AND t.prihlasit_od < NOW() AND t.prihlasit_do > NOW()AND a
      .id_akad_rok = ? AND z.id_akad_rok = ? AND z.id_uzivatel = ? AND t.prihlas = 1 AND t.id_atermin =
      ? AND t.konec > NOW() AND a.id_aktivity NOT IN ( SELECT ak.id_aktivity FROM tu_aktivita AS ak
    INNER JOIN tu_atermin AS at ON ak.id_aktivity = at. id_aktivity
    INNER JOIN tu_xstskup AS xs ON at.id_skupina = xs.id_skupina
    WHERE at.konec > NOW() AND xs.id_student = ?)
```

5. Přihlášení studenta k testu
   5.1 Začátek transakce
   5.2 Kontrola, zdali je kapacita termínu testu *id_atermin* stále volná z tabulky **tu_atermin**

```
SELECT COUNT(*), t.kapacita FROM tu_atermin AS t
  LEFT JOIN tu_xstskup AS x ON t.id_skupina = x.id_skupina
  WHERE t.id_atermin = ? GROUP BY t.id_atermin
  HAVING COUNT(x.id_stskup) < t.kapacita
```

   5.3 Výběr aktuální skupiny termínu *id_skupina* z tabulky **tu_atermin** s id *id_atermin*

```
SELECT id_skupina FROM tu_atermin WHERE id_atermin = ?
```

   5.4 Vložení přihlášení k termínu do tabulky **tu_xstskup** (atributy *id_skupina*, *id_uziv*, *id_uziv*)

```
INSERT INTO tu_xstskup (id_stskup, id_skupina, id_student, kdo_zme) VALUES (',?,?,?)
```

   5.5 Dekrementace počtu vygenerovaných testů a inkrementace počtu hotových **tu_aktivita**

```
UPDATE tu_aktivita SET poc_vygen = poc_vygen−1, poc_hotov = poc_hotov+1 WHERE id_aktivity = ?
```

5.6 Konec transakce

6. Zobrazení hlášení o přihlášení k termínu testu

Rozšíření

4a Pokud termín není studentovi z nějakého důvodu přístupný, vypíše se studentovi hlášení *„Nelze přihlásit. Termín není dostupný.“* a pokračuje se krokem 1.

5.2a Pokud je termín již plný, transakce se ukončí, vypíše se studentovi hlášení *„Nelze přihlásit. Termín je již plný.“* a pokračuje se krokem 1.

UC24: Spuštění testu

Aktér	systém
Úroveň	uživatelská
Spouštěč	student spustí ostrý test
Vstupní podmínky	identifikátor aktivity *id_aktivity*

Hlavní scénář

1. Zobrazení seznamu testů výběrem aktivních testů ke spuštění (aktivita je test, student má termín testu zapsaný a tedy existuje v dané skupině testu, daný termín ještě není ukončený, daný test je již spuštěn anebo je aktuální čas spuštění v daném časovém intervalu termínu) z tabulky **tu_aktivita** a zbylých tabulek uvedených v dotazu (vstupní atributy jsou *id_uziv*, *id_uziv*, *id_akad_rok*, *id_akad_rok*, *id_uziv*):

```
SELECT a.id_aktivity, a.po_jedne, t.nazev AS predmet, a.dobaA AS cas, at.mistnost, DATE_FORMAT(at.
    zacatek,'%H:%i %d.%m.%Y') AS zacatek, r.popis AS semestr, 'Spustit' AS akce FROM tu_aktivita
    AS a
  INNER JOIN tu_tpredmet AS t ON a.id_tpred = t.id_tpred
  INNER JOIN tu_zapis AS z ON a.id_tpred = z.id_tpred
  INNER JOIN tu_akad_rok AS r ON a.id_akad_rok = r.id_akad_rok
  INNER JOIN tu_atermin AS at ON a.id_aktivity = at.id_aktivity
  INNER JOIN tu_xstskup AS xs ON at.id_skupina = xs.id_skupina
  LEFT JOIN tu_astud AS st ON at.id_atermin = st.id_atermin
  WHERE xs.id_student = ? AND z.id_uzivatel = ? AND z.id_akad_rok = ? AND a.id_akad_rok = ? AND a.
      id_typAktivity = 'TypA000002' AND st.datum_do IS NULL AND ((at.zacatek < NOW() AND at.
      konec > NOW()) OR st.id_student = ?)
```

2. Systém zobrazí tabulku uživateli, obsahující seznam aktivních testů se sloupci *nazev*, *predmet*, *semestr*, *mistnost*, *zacatek*, *cas*, *akce* (odkaz pro spuštění)

3. Student spustí test zvolením odkazu „Spustit“ u vybraného testu

4. Kontrola, pokud není spuštěn jiný test či autotest než vybraný (ze session)

5. Kontrola, pokud má student práva k zobrazení testu ověřením z tabulky **tu_aktivita**

6. Kontrola odkazu, zdali se nemají zobrazit pokyny

7. Kontrola, pokud ještě není spuštěn daný test - neexistuje v session (je aktuálním testem)

8. Výběr testu

   8.1 Kontrola, zdali v databázové tabulce **tu_gstruk** jsou pro danou aktivitu (test), vygenerované seznamy otázek a odpovědí

8.2 Výběr testovacích vrstev testu (seznam *idSstrList*) a nově vytvořené *id_astud* aktuálního termínu testu

8.2.1 Začátek transakce

8.2.2 Výběr všech vygenerovaných testů pro danou aktivitu *id_akti* z tabulky **tu_gstruk**

```
SELECT DISTINCT(id_testu) FROM tu_gstruk WHERE id_akti = ?
```

8.2.3 Výběr prvního 1 náhodného testu (číselné id *id_testu*) z vybraných testů

8.2.4 Výběr struktury zvoleného testu *id_testu* z vybraných testů (vstupem jsou proměnné *id_aktivitity*, *id_uziv*, *id_testu*)

```
SELECT gs.id_gstr, gs.id_tvrs, at.id_skupina FROM tu_gstruk AS gs INNER JOIN tu_atermin AS at ON gs.
    id_akti = at.id_aktivity
  INNER JOIN tu_xstskup AS xs ON at.id_skupina = xs.id_skupina
  WHERE at.id_aktivity = ? AND xs.id_student = ? AND gs.id_testu=? AND at.zacatek < NOW() AND at.
      konec > NOW()
  ORDER BY gs.poradi ASC
```

8.2.5 Vytvoření nového číselného id *noveId* pro uložení testu studenta z tabulky **tu_sstruk**

```
SELECT MAX(id_testu)+1 FROM tu_sstruk
```

8.2.6 Pro každou vybranou vrstvu se provede:

8.2.6.1 Výběr samotných odpovědí z tabulky *tu_godpov* podle *id_gstr*

```
SELECT id_odpo, poradi_go FROM tu_godpov WHERE id_gstr = ? ORDER BY poradi_go ASC
```

8.2.6.2 Vytvoření jedné otázky z otázky *id_gstr* do **tu_sstruk** voláním uložené procedury *tu_sstruk_insert*, která vrací ID vytvořeného záznamu *id_sstruk* (vstupem je *id_akti*, *id_tvrs*, *id_uziv*, *id_skupina*, *noveId*, *id_uziv*)

```
SELECT tu_sstruk_insert(?,?,?,?,?,?)
```

8.2.6.3 Pro každou odpověď se vytvoří záznam v tabulce **tu_sodpov** (vstupem je *id_sstruk*, *id_odpo*, *poradi_go*, *id_uziv*)

```
INSERT INTO tu_sodpov (id_sodp, id_sstr, id_odpo, poradi_so, kdo_zme) VALUES (',?,?,?,?)
```

8.2.6.4 Smazání vybraných záznamů z tabulky **tu_gstruk** a **tu_godpov** podle *id_aktivity* a *id_testu* (uložený trigger odstraní nejdříve odpovědi k dané vygenerované otázce)

```
DELETE FROM tu_godpov WHERE id_gstr = OLD.id_gstr;
DELETE FROM tu_gstruk WHERE id_akti = ? AND id_testu = ?
```

8.2.7 Vytvoření záznamu obsahující spuštěné testy do tabuky **tu_astud**

8.2.8 Konec transakce

8.3 Uložení aktuálního spuštěného testu (pole všech otázek *id_sstr*), času spuštění a *id_astud* do session. To je následně uloženo do osobního uživatelského nastavení

v tabulce **sy_unastaveni**
8.4 Uložení záznamu o přihlášení k testu do protokolu (tabulka **ex_protokol**)
9. U aktuálního testu se aktualizuje IP adresa s názvem počítače studenta v **tu_astud**
10. Výběr identifikátorů testovacích vrstev *id_tvrs* podle pole *idSstrList* z tabulky **tu_sstruk**, které se uživateli poté zobrazí
11. Výběr názvu testu do titulku z tabulky **tu_aktivita**

Rozšíření
2a Pokud se jedná o test zobrazený se všemi otázkami najednou v původním okně (*po_jedne* je false), odkaz pro spuštění se nastaví na Student:MujTestOkno:zobraz
2b Pokud se jedná o test zobrazený v novém okně po jedné otázce (*po_jedne* má hodnotu true), odkaz pro spuštění se nastaví na Student:MujTest:zobraz
2c Pokud se mají zobrazit pokyny (kontrolou v tabulce **tu_xkomponenta**), nastaví se do odkazu
4a Pokud je spuštěn jiný test či autotest, zobrazí se informace o nedostupnosti a test se nezobrazí
5a Pokud student nemá práva zobrazit test, nelze test zobrazit
6a Pokud se mají zobrazit pokyny testu, provede se redirect na zobrazení pokynů
7a Pokud již daný test je spuštěn, pokračuje se krokem 9.
8.1a Pokud testy vygenerované nejsou, zavolá se funkce modulu tutor pro generování nových
8.2.6.3a Pokud odpověď neexistuje (pro tvořenou otázku), je vložen prázdný řetězec
8.2.7a V případě, že záznam v **tu_astud** již existuje, aktualizují se změněné atributy
9a Pokud se změnila IP adresa nebo název počítače, inkrementuje se ještě počet změn IP

```
UPDATE tu_astud SET ip_pocitac = ?, nazev_pocitac = ?, kdo_zme = ?, pocet_zmen_ip = pocet_zmen_ip + 1
    WHERE id_astud = ?
```

UC25: Uložení testu

Aktér	systém
Úroveň	uživatelská
Spouštěč	uživatel stiskne tlačítko „Uložit test“ při spuštěném testu
Vstupní podmínky	spuštěný vygenerovaný test

Hlavní scénář
1. Validace
    1.1 Kontrola, zdali existuje identifikátor aktivity v přijatých datech z response
    1.2 Kontrola, zdali je test (aktivita) v aktuálních aktivitách v session
    1.3 Kontrola, zdali je seznam otázek v otázkách aktuální aktivity v session
    1.4 Vyfiltrování identifikátorů *id_sstr*, které obsahují doplňovačku a kontrola formátu odpovědí všech těchto doplňovaček
2. Kontrola, zdali se může test uložit a čas ukončení je v rozmezí testu s 10 vteřinovou rezervou (provádí se podle času spuštění uloženého v session)
3. Úprava dat z doplňovaček pro každou vrstvu

3.1 Odpovědi se přetransformují do vektoru slov *vektor*
3.2 Uložení nastavení vrstvy s doplňovačkou *vektor* do databázové tabulky **st_ntvrstva**
4. Odeslání a provedení akce uložení každé otázky
4.1 Uložení zodpovězených variantních odpovědí do tabulky **tu_sodpov** se vstupem *id_uziv*, *id_sstr* a polem zodpovězených odpovědí *id_odpo*

```
UPDATE tu_sodpov SET obsah_so = 1, kdo_zme = ? WHERE id_sstr=? AND id_odpo IN %in
```

4.2 Uložení tvořených odpovědí do **tu_sodpov** se vstupem: text odpovědi, *id_uziv*, *id_sstr*

```
UPDATE tu_sodpov SET obsah_so = ?, kdo_zme = ? WHERE id_sstr=?
```

4.3 Uložení nezodpovězených variantních odpovědí do tabulky **tu_sodpov**

```
UPDATE tu_sodpov SET obsah_so = 0, kdo_zme = ? WHERE id_sstr=? AND id_odpo NOT IN %in
```

4.4 Protokol
4.4.1 Výběr potřebných dat *data* pro protokol z tabulky **tu_sodpov** a **tu_sstruk**
4.4.2 Uložení záznamu s údaji *data* pro každou otázku popisující přihlášení k odpovědi do protokolu (tabulka **ex_protokol** a **ex_vyhodnoceni**)
5. Zobrazení informačního hlášení „Uloženo!“

Rozšíření
1.1a Pokud id_aktivity v response neexistuje, nelze provést uložení a zobrazení testu
1.2a Pokud není test v aktuálních (v session), nelze provést uložení a zobrazení testu
1.3a Pokud se seznam otázek v session nevyskytuje, nelze provést uložení a zobrazení testu
1.4a Pokud odpovědi doplňovaček mají nesprávnou strukturu, nelze provést uložení a zobrazení testu
2a Pokud je provedeno odevzdání po limitu i po 10 vteřinové rezervě, vypíše se hlášení *„Test nelze uložit. Již je po limitu.“*

4.3 Návrh implementace

Kapitola obsahující části systému a jejich popis vytvořené při vývoji LMS Barborka 4 v etapě návrhu implementace.

4.3.1 Struktura Systému

Celková struktura systému a návrh presenterů vyplývá z možnosti Nette vytvářet hierarchie presenterů, z požadavků pro menu a z potřeby jednoznačně definovat elementární funkce. Proto byla navržena struktura elementárních funkcí, kdy jeden presenter zastupuje funkcionalitu jedné elementární funkce. Tímto jsou zajištěny společné elementární funkce, které se ve stromové struktuře menu (funkcí) vyskytují na různých místech, ale provádějí stejnou funcionalitu. Obě položky menu se odkazují na jeden společný presenter vykonávající požadované akce. Podle teorie adaptivní výuky je systém rozdělen

na několik subsystému. Díky možnostem Nette lze systém takto jednoduše rozdělit na moduly.

Adresářová struktura systému rovněž logicky vychází z Nette. Moduly se nacházejí v dílčích složkách a jednotlivé vrstvy architektury MVC v dalších pod-složkách (models, presenter, templates) [8]. Odděleny od této struktury jsou dokumentace, manuály, prototypová verze systému, jednotkové testy a soubory obsahující vytvoření databáze se systemovými i testovacími daty. Ty obsahují SQL příkazy pro definici a pro vkládání dat.

V datové vrstvě se pracuje se dvěma typy připojení k databázi. Obě technologie byly vyvinuty autory Nette. Jedná se databázové vrstvy dibi a Nette\Database. Druhá knihovna je navržena pro práci s tabulkami vycházejícími z NotORM [26]. Pro vývoj systému byly zvoleny dva přístupy, které je možné použít a kombinovat libovolně. Prvním přístupem je použití zabudovaných metod obou připojení bez potřeby psát SQL dotazy. Druhým pro některé pohodlnějším přístupem vytváření vlastních dotazů rovněž za použití daných připojení. U transakcí je potřeba použít pouze jedno databázové připojení.

Pro snadné jednotkové testování naimplementovaných metod presenterů byla do systému ve vrstvě model zavedena technika dependency injection pro jednotlivé třídy. Jedná se o možnost testování částí kódu bez závislosti na databázi. V podstatě dochází k náhradě celých tříd modelu za třídy obsahující definovaná testovací data, která presenter používá, aniž by se cokoliv jiného změnilo.

Systém mimo jiné používá i služby dostupné v každém presenteru (přes context[9]). Jedná se vesměs o služby vykonávající velmi časté funkcionality spojené s jádrem systému, jako například databáze, protokol, globální nastavení, soubory, slovník, autentifikátor, autorizátor, virtuální učitel a jiné. Bezpečnost systému je opět zajištěna funkcemi Nette, jako je ochrana proti různým útokům (například cross-site scipting, sql injection, útoky na session atd.) a zabudované funkce pro automatickou validaci formulářů [27].

4.3.2 Návrh indexů

Výchozí databázové indexy jsou vytvářeny automaticky databázovým systémem MySQL u atributů všech tabulek, které jsou primární klíče, cizí klíče či unikátní. Výchozí indexy nejsou zmíněny v níže uvedené tabulce 5. Tabulka obsahuje indexy navržené z typických SQL dotazů a nejčastějších podmínek dotazů. To urychlí proces výuky a testování studentů. Některé z těchto indexů jsou složené. Navržené indexy vychází z atributů i jiných tabulek, než obsahuje studentská či administrátorská databáze, z důvodů zrychlení výuky a o to se jedná především.

Závěrečná optimalizace systému a s ní spojený návrh složitějších optimalizačních databázových struktur bude ovšem provedena následovně po celkovém otestování a po zjištění možné nedostatečně rychlé odezvy při ostrém provozu.

[8] Uvedené rozdělení modulů je dostupné na http://doc.nette.org/cs/presenters#toc-moduly

[9] Context je systémový kontejner Nette umožňující přístup ke službám a modelům

Název indexu	Tabulka	Atributy	Unikátní
st_ntvrstva_uzi_rez_tvr_idx	st_ntvrstva	id_uziv, rezim, id_tvrstva	ANO
sy_menu_volani_m_idx	sy_menu	volani_m(50)	
sy_menu_zobrazit_m_idx	sy_menu	zobrazit_m	
sy_unastaveni_naz_uzi_n_idx	sy_unastaveni	nazev_n ,id_uziv	ANO
sy_zpristupneni_zac_kon_n_idx	sy_zpristupneni	zacatek_z, konec_z	
tu_aktivita_typ_aka_akt_idx	tu_aktivita	id_typAktivity, id_akad_rok, id_aktivity	
tu_astud_stav_as_idx	tu_astud	stav_as	
tu_astud_datum_do_idx	tu_astud	datum_do	
tu_atermin_zod_zdo_idx	tu_atermin	zobrazit_od, zobrazit_do	
tu_atermin_zac_kon_idx	tu_atermin	zacatek, konec	
tu_atermin_konec_idx	tu_atermin	konec	
tu_atermin_pod_pdo_idx	tu_atermin	prihlasit_od, prihlasit_do	
tu_tzpristupneni_zac_kon_idx	tu_tzpristupneni	konec_z, zacatek_z	
tu_tpredmet_pri_tpr_idx	tu_tpredmet	prihlasovani, id_tpred	
au_ctotaz_ftyp_o_idx	au_ctotaz	typ_o(1)	
au_vrstva_poradi_vr_idx	au_vrstva	poradi_vr	
au_tvrstva_poradi_tvr_idx	au_tvrstva	poradi_tvr	
au_ramec_por_lek_idx	au_ramec	poradi_r, id_lekce	
au_lekce_por_pre_idx	au_lekce	poradi_l, id_pred	
au_odpoved_tvr_spr_idx	au_odpoved	id_tvrstva, sprav_o	

Tabulka 5: Návržené indexy

4.3.3 Návrhové vzory

Podle navrženého MVC vzoru frameworku Nette je systém rozdělen na tři základní vrstvy. Tou první je model, který představuje datový a funkční základ celé aplikace, presenter zastávající roli controlleru, tedy řadiče zpracovávajícího požadavky uživatele, na jejichž základě volá patřičnou aplikační logiku, a šablony, které zobrazují uživateli výsledek jeho požadavku [28]. Pro tento případ byla část aplikační logiky zařazena do presenterů kvůli snadnější práci s metodami obsaženými v navržené hierarchické struktuře presenterů.

Při dalším návrhu tříd a komponent systému se postupovalo podle různých základních návrhových vzorů. Nejčastěji se jedná o vzor party, který definuje zobecnění společných metod a členských proměnných provádějících danou funkcionalitu. Tento přístup je zaveden v hierarchii presenterů. Dále je použit vzor továrna u testování stu-

dentů v okně a uvnitř výuky. Testování je rozděleno na autotesty a testy. Ty mají shodnou strukturu, formu vytváření a zobrazování, ale používají příslušné modely. Také je použit základní hierarchický organizační vzor pro objektovou reprezentaci položek menu.

V příloze se nachází 4 vybrané třídní diagramy, obsahující schémata struktury menu, výuky, testování a základního jádra systému. Obsahují rovněž vazby na třídy úzce související s danou funkcionalitou. V třídním diagramu popisujícím menu se vyskytuje hlavní abstraktní MenuPresenter, který obsahuje komponenty a dané datové entity pro zobrazení jednotlivých částí menu. Všechny presentery, jež souvisí se zobrazením elementární funkce se základní nabídkou, musí dědit tento presenter. Druhý diagram pro zobrazení výuky znázorňuje použití komponent pro jednotlivé části výuky. V tomto případě je také znázorněno použití třídy, obsahující funkce s generováním výukových komponent, pro toto použití je třeba implementovat rozhraní ISpolecneKomponenty.

Další třídní diagram zobrazuje celou strukturu tříd vytvořenou pro testování studentů. V této části bylo potřeba navrhnout abstraktní třídy, vykonávající společné funkcionality testů/autotestů a také rozlišit funkcionalitu pro generování testování v novém okně po jedné otázce či se všemi otázkami najednou. Jednotlivé presentery ve spodní části hierarchie definují parametry testování a model, který se v aktuálním případě používá. K tomuto modelu přistupují všechny metody abstraktních rodičovských tříd vykonávající různé funkcionality. U presenterů o úroveň výše je rozlišeno, zdali se jedná o testování po jedné otázce či všech najednou, a ty obsahují příslušné komponenty. U elementárních funkcí zobrazujících seznam aktivních testů a autotestů v tabulce se v podstatě jedná o stejný návrh. V diagramu jsou také znázorněny obecné třídy, obsahující generování vyhodnocení a metadat výukových komponent. V posledním diagramu se nachází třídy, které mají úzkou souvislost s presenterem BasePresenter a základními funkcemi systému (globální proměnné a uživatelské nastavení, slovník, autentifikace a autorizace).

4.3.4 Popis tříd a rozhraní

4.3.4.1 Modul Student

ATModel – Abstraktní třída, definující společné metody modelů testování (testy, autotesty), obsahuje společné metody pro výběr dat z databáze a transformuje data do příslušné podoby.
AutoTestyModel – Práce s daty potřebnými pro generování autotestů.
MePredmetyModel – SQL dotazy zaměřující se na výběr studentových zapsaných předmětů.
MeTestyModel – Metody sloužící pro výběr dat potřebných k zobrazení aktivních testů aktuálního studenta.
MeUkolyModel – Model obsluhující dotazy pro všechny studentovy úkoly - historické, aktuální i budoucí (záznamy tabulky tu_astud).
NastaveniTVrstvy – Datová entita nesoucí potřebné informace o jednom uživatelském nastavení a stavu testovací vrstvy.
PrihlaseniPredmetModel – Model, který obsahuje dotazy společné se zápisem (přihlášením) studenta k danému předmětu.
PrihlaseniTestModel – Logika přihlášení k danému termínu testu.

StavTVrstevModel – Služba obsahující ukládání a načítání studentských nastavení vyplněných a zodpovězených testovacích vrstev.
TestyModel – Práce s daty potřebnými pro výběr a generování testů.
VerejnePredmetyModel – Výběr veřejných předmětů z databáze.
VyukaModel – Model obsahující práci s databázovými daty týkající se celkové výuky.
AMTOPresenter – Abstraktní třída obsahující společné funkce pro zobrazení testu po jedné otázce v novém okně, jeho zpracování a uložení.
AMTPresenter – Abstraktní třída obsahující společné funkce pro zobrazení testu najednou uvnitř systému v původním okně, jeho zpracování a uložení.
ATControl – Abstraktní komponenta, v níž se nachází všechny společné metody určené pro zobrazení a vykreslení testů v jakémkoliv formátu testování (po jedné, najednou). Definuje strukturu, jakou dané komponenty musí mít.
ATestovaniPresenter – Abstraktní třída obsahující společné funkce pro testování – generování testů, autotestů, seznamů testů, zobrazení pokynů či samotného zobrazení testu.
ATPresenter – Abstraktní třída obsahující společné funkce pro zobrazení seznamu aktivních autotestů či testů v tabulce.
AutotestyPresenter – Presenter zobrazující seznam aktivních autotestů.
AVyhodnoceniPresenter – Abstraktní třída obsahující společné funkce pro vyhodnocení všech testů či autotestů a následného zobrazení studentovi.
MujAutotestOknoPresenter – Definice parametrů a modelu pro autotest zobrazený studentovi v novém okně po jedné otázce zvlášť.
MujAutotestPresenter – Presenter, jenž obsahuje parametry a model pro zobrazení autotestu studentovi najednou v původním okně.
MujTestOknoPresenter – Tento presenter obsahuje parametry a model určující test zobrazený studentovi v novém okně po jedné otázce zvlášť.
MujTestPresenter – Presenter, který obsahuje parametry a model pro zobrazení testu studentovi najednou v původním okně.
TestVrstvyControl – Komponenta generující a zobrazující vrstvy pro testování najednou.
TestVrstvyOknoControl - Komponenta generující a zobrazující vrstvy pro testování v okně.
TestyPresenter – Presenter zobrazující seznam aktivních testů.
NticeControl – Komponenta generující a zobrazující ovládací prvek výuky - 12tici typů variant a hloubek dostupných k aktuální zvolené vrstvě.
ObsahPresenter – Zobrazení samotného obsahu výukové komponenty v novém okně.
VrstvyControl – Komponenta, která má na starost převod databázové struktury s vrstvami na strukturu zobrazitelnou v příslušné šabloně a následné zobrazení všech vrstev výuky.
VyukaPresenter – Hlavní funkcionalita procesu výuky spojena s výběrem a zpracováním jednotlivých částí výuky, generováním výuky, komunikací s ostatními moduly pro získání AVS, reakcí na odpovědi či vyhodnocení testovacích otázek.
ISpolecneKomponenty – Rozhraní pro presentery a komponenty používající třídu pro zobrazení výukových komponent.
MePredmetyPresenter – Presenter zobrazující studentem zapsané předměty.
MeTestyPresenter – Presenter obsahující správu všech vykonaných i nevykonaných testů studenta s možností zobrazit vyhodnocení.

MeUkolyPresenter – Správa všech úkolů studenta včetně zobrazení diskuze pomocí komponenty DiskuzeControl.
PredmetyPresenter – Abstraktní presenter obsahující společné modely použité pro zobrazení seznamu předmětů v modulu student.
PrihlaseniPresenter – Presenter, jehož úkolem je provedení zápisu uživatele do předmětu.
SpolecneKomponenty – Třída obsahující společné metody pro generování struktury z metadat komponent použité při zobrazení výukových komponent.
SpolecneVyhodnoceni – Třída obsahující společné metody, které obstarávají převod databázových dat na strukturu využitelnou pro zobrazení vyhodnocení v šabloně.
TestPrihlaseniPresenter – Funkce vykonávající přihlášení studenta k danému termínu.
VerejnePredmetyPresenter – Presenter zobrazující veřejné předměty.

4.3.4.2 Modul Administrátor

ModelDatabazeModel – Výběr dat ze systémového katalogu databáze LMS Barborka 4.
CTextModel – Model obsahující funkcionalitu, spojenou s výběrem a správou typů slovních spojení slovníku z databázového číselníku.
SlovnikModel – Model pro výběr slovníkových dat z databáze pro SlovnikPresenter.
DetailPolozky – Objektová datová entita reprezentující údaje o položce menu v elementární funkci tvorby menu, obsahující detail zobrazené položky.
TvorbaMenuModel – Model zaštiťující komunikaci mezi databází a systémem. Dochází k vygenerování struktury s objekty třídy DetailPolozky.
UzivateleModel – Třída obsahující metody pro správu všech uživatelů systému a komunikaci s příslušnou částí databáze obsahující uživatele.
HistorieModel – Model, obsahující metody pro výběr minulých registrací do systému.
ZpristupneniModel – Model vázaný na ZpristupneniPresenter zprostředkovává databázová data pro zpřístupnění registrace do systému.
GlobalniPromennePresenter – Správa globálních proměnných systému.
ModelDatabazePresenter – Metody pro zobrazení modelu databáze (celkového seznamu všech databázových tabulek a jednotlivých tabulek s atributy).
SlovnikPresenter – Presenter obsluhující funkce administrátora pro správu slovníku.
TvorbaMenuPresenter – Obsluha požadavků spojených s tvorbou a úpravou hierarchické struktury menu a elementárních funkcí systému.
TypySlovnichSpojeniPresenter – Presenter obsluhující elementární funkci administrátora pro slovník – typy slovních spojení.
UzivatelePresenter – Hlavní presenter, který obsluhuje správu všech uživatelů a jejich hromadný import do systému.
ZpristupneniPresenter – Presenter, jenž má na starost elementární funkci administrátora pro správu aktuálních zpřístupnění a zobrazení historie registrace studentů do systému.

4.3.4.3 Řídicí systém

GeneratorMenu – Obecně použitelná komponenta zobrazující levé menu z hierarchické objektové struktury do příslušné šablony.
PolozkaLevehoMenu – Objektová reprezentace položky levého menu, je používána pro ge-

nerátor levého menu. Obsahuje pole podpoložek stejné třídy.
MenuModel – Model pro generování menu, obsahuje metody pro vygenerování hierarchické struktury menu získané z databáze.
Back – Objektová návratová hodnota pro rekurzivní nastavení aktuálního a zvýrazněného uzlu předka u generování struktury menu.
Obsah – Objektová reprezentace položky hlavního menu (u menu s moduly či submenu).
Authenticator – Třída vykonávající autentizaci uživatelů LMS Barborka 4.
Authorizator – Třída zprostředkovávající autorizaci uživatelů systému k jednotlivým elementárním funkcím.
Nastaveni – Služba systému, která obsluhuje globální proměnné systému, osobní uživatelské proměnné a jejich ukládání do session a databáze.
Prispevek – Datová entita reprezentující jeden příspěvek v diskuzi.
Protokol – Služba, která vykonává ukládání jednotlivých akcí do protokolu v databázi.
Slovnik – Slovník systému sloužící pro překlad slovních spojení do cizího jazyka.
BasePresenter – Presenter obsluhující základní společné funkce jádra systému.
CountDownControl – Komponenta zobrazující odpočítávání času a provádějící synchronizaci podle vlastního vloženého času.
DiskuzeControl – Komponenta zobrazující diskuzi s jednotlivými příspěvky (třída Prispevek).
HlavniMenuControl – Komponenta, která zaštiťuje vygenerování položek do objektové struktury (Obsah) a zobrazení celkového hlavního menu.
LeveMenuControl – Komponenta zobrazující levé menu pomocí generátoru menu (GeneratorMenu), pracující se strukturou vytvořenou z instancí třídy PolozkaLevehoMenu.
MenuControl – Abstraktní komponenta, která obsahuje společné modely pro zobrazení menu.
MenuPresenter – Abstraktní presenter provádějící automatické zobrazení menu. Všechny třídy (presentery), které vykonávají elementární funkci spojenou se zobrazením menu, musí vždy rozšiřovat tento presenter.
NastaveniPresenter – Společné metody pro uživatelská nastavení, zatím pouze pro změnu údajů a hesla.
LDAP - Služba pro komunikaci se školním serverem pomocí LDAP.
SouborModel – Služba pro ukládání souborů s daty do adresářové struktury na serveru.
AUzivatelePresenter – Abstraktní třída pracující se správou školních údajů u studentů (škola, fakulta, obor) a zaměstnanců (škola, fakulta, katedra) a načítáním do selectboxů přes AJAX.
NeimplementovanoPresenter – Zobrazení obrazovky s upozorněním o spuštění elementární funkce systému, která zatím nebyla implementována.
NelzeZobrazitPresenter – Zobrazení obrazovky s upozorněním o nedostupnosti uživatelem spuštěné funkce, akce či funkčnosti.
PrihlaseniPresenter – Presenter obsluhující celý proces přihlášení uživatele do systému.
RegistracePresenter – Proces registrace studentů do systému v době zpřístupnění.
SouborPresenter – Presenter obsluhující požadavky na všechny uložené soubory.

4.4 Implementace

Z požadavků, analýzy a návrhu byly implementovány všechny zmíněné funkčnosti uvedené v seznamech. V poslední části této práce jsou pro ukázku stručně popsány a zobrazeny významné řádky zdrojových kódů, které jsou definované ve frameworku Nette, jazyce PHP anebo vlastně vytvořené. Popis vytvořených komponent a celkového použití dostupných metod je uveden v programátorské příručce. Nakonec jsou zde zobrazeny ukázky systému.

Součástí základní implementace jsou triggery, procedury a funkce vytvořené na straně databáze z důvodů zrychlení vykonávaní dotazů a aktualizací. Klíče databázových tabulek nejsou automaticky generovány databázovým systémem kvůli složitější struktury. Bylo potřeba na straně databáze vytvořit proceduru, která zajistí inkrementaci nově vytvářeného záznamu. Tuto proceduru volají triggery navázané na každou tabulku udávající prefix klíče. Dále jsou v databázi vytvořeny procedury, pracující s položkami menu a jejich úpravou (vkládání, mazání, řazení) a další.

4.4.1 Významné řádky zdrojových kódů

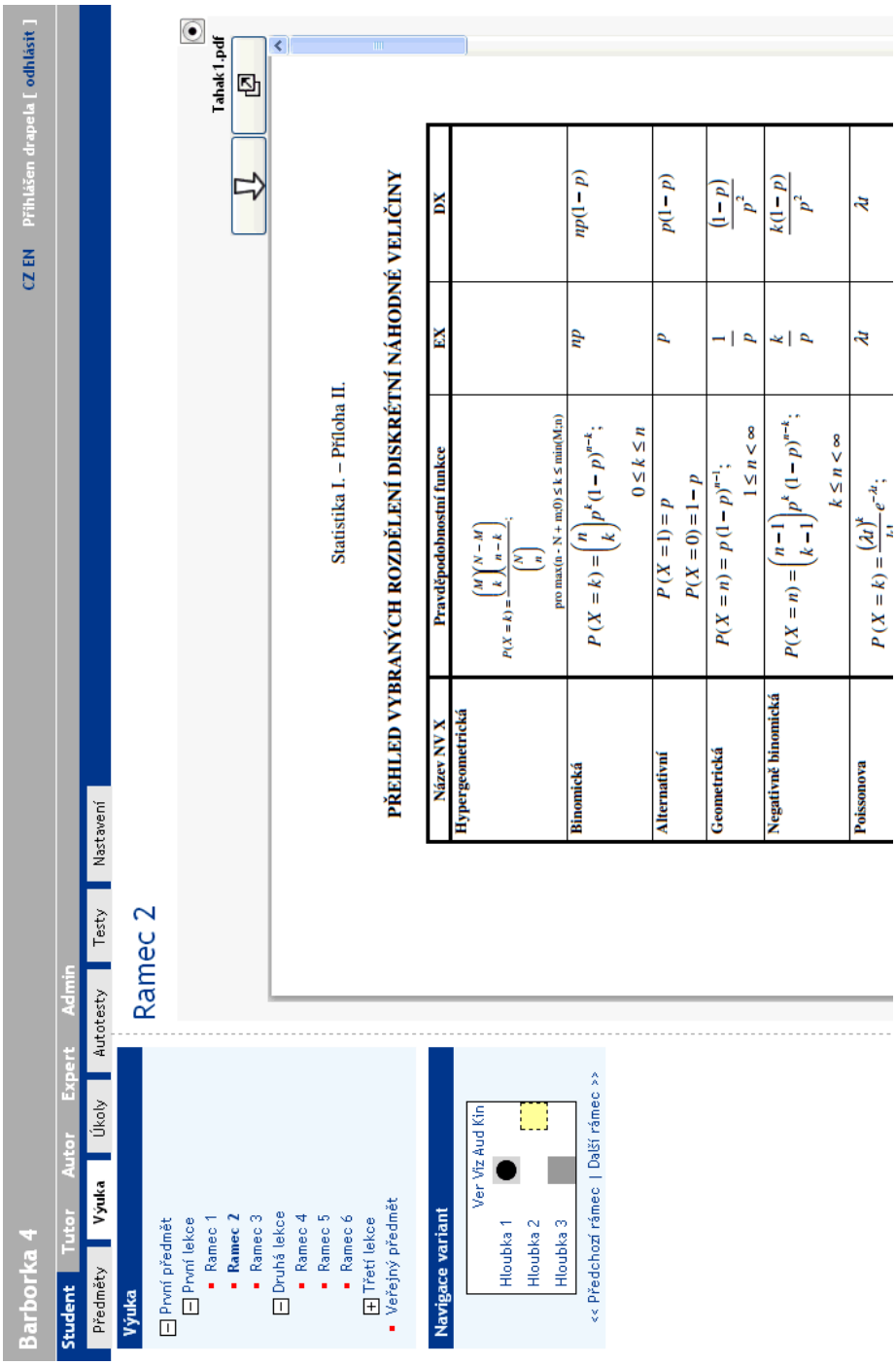
Výběr aktuálního uživatele systému pomocí metody Nette.
```
$this->getUser();
```
Automatické přihlášení do systému metodou Nette
```
$user->login($login, $heslo);
```
Odhlášení ze systému včetně odstranění identity metodou Nette
```
$user->logout(TRUE);
```
Nastavení expirace přihlášení uživatele a session stejnou hodnotou
```
$user->setExpiration($expirace, TRUE, TRUE);
$session->setExpiration($expirace);
```
Přidání chyby formuláře pro vypsání při odeslání
```
$form->addError('Chyba');
```
Vypsání informačního hlášení daného typu
```
$this->flashMessage('Hlášení', 'typ');
```
Nastavení událostí přiřazením odkazu na funkci do pole (v tomto případě formuláři do pole pro úspěšné odeslání formuláře)
```
$form->onSuccess[] = callback($this, 'odesliFormular');
```
Nastavení vlastního slovníku (ve tvaru služby) dané komponentě
```
$component->setTranslator($this->context->slovnik);
```
Přeložení výrazu do aktuálního jazyka mimo strukturu presenterů (volání statické metody)
```
\BaseModule\Slovnik::preloz("Výraz");
```
Nastavení šablony pro zobrazení akce pomocí metody Nette
```
$this->setView("zobraz");
```
Serializace a deserializace proměnné či pole
```
$data = base64_encode(serialize($promenna));
$promenna = unserialize(base64_decode($data));
```

Rozdělení řetězce obsahující doplňovačku na jednotlivé části
```
$casti = preg_split("/(<#)|(#>)/", $obsah_k);
```
Lambda funkce spočívající v odstranění hvězdičky na začátku textu každého slova v poli
```
$odstraneniHvezdicky = function($value)
{ return ($value[0] == '*')?substr($value, 1):$value; };
```
Vložení šablony pro zobrazení výukové komponenty
```
{include '../Vyuka/komponenta.latte', nazev => $nazev, obsah_k =>
$obsah_k, typ_ck => $typ_ck, mime => $mime, cesta => $cesta}
```
Dynamické vytvoření komponenty pro zobrazení formuláře s odpověďmi
```
{control "formularTvoreneOdpovedi-" . $vrstva ['vlastnosti']
['id_tvrstva']}
```
Dynamické vytvoření formuláře obsahující dynamicky vygenerované části textu
```
{form "formularDoplnovacka-" . $vrstva ['vlastnosti']
['id_tvrstva']}
```
Příklad použití volání uložené procedury na straně databáze
```
$this->db->query('CALL sy_menu_insert(?, ?, ?, ?, ?, ?, ?, ?, ?)',
$rodic, $id, $text, $popis, $napoveda, $odkaz, $poradi, $zobrazit,
$kdo_zme);
```
Validace emailu
```
$validni = filter_var($email, FILTER_VALIDATE_EMAIL);
```

4.4.2 Ukázky aplikace

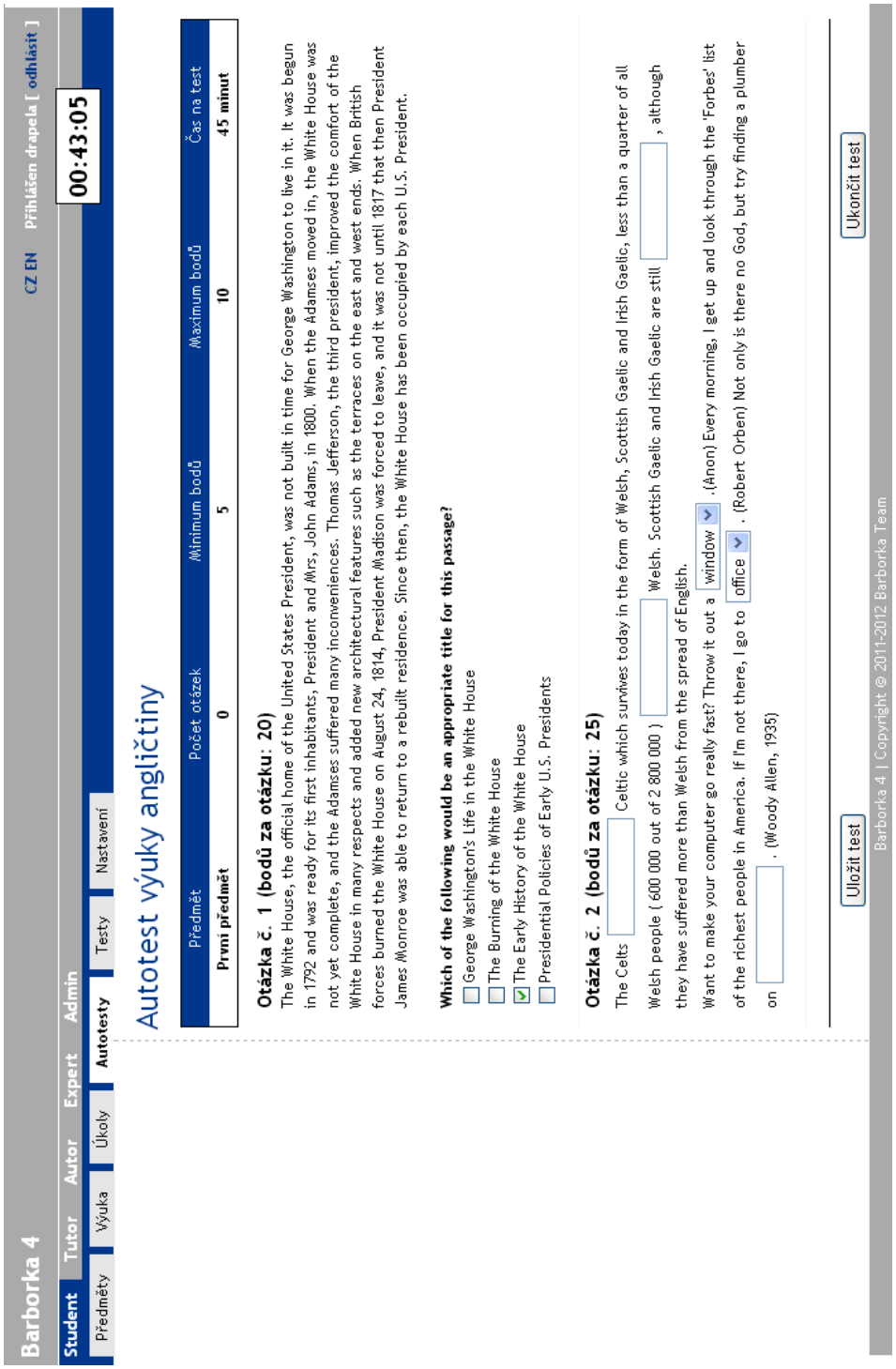
V této části jsou zobrazeny sejmuté obrazovky ze zkušebního provozu LMS Barborka 4. Jsou zde uvedeny obrázky s ukázkami výuky (obrázek 19), autotestu (obrázek 20), modelu databáze (obrázek 21) a detailu úkolu (obrázek 22). Ve výuce jsou zobrazeny navigační prvky nacházející se v levé části (navigační levé menu, 12tice, další rámec a následující rámec). V pravé části je zobrazena vrstva obsahující komponentu - PDF soubor s „multibuttonem". Obrázek zobrazující testování obsahuje vygenerovaný autotest se všemi otázkami najednou. Je zde vidět odpočítávání času, hlavička autotestu, samotné otázky s odpověďmi, tlačítko pro uložení a pro ukončení. V obrázku obsahující databázovou tabulku v rámci modelu databáze je zobrazena samotná tabulka s atributy a klíči. Dále je zde zobrazen lineární zápis a závislé tabulky. Poslední obrázek obsahuje detail testu s jednotlivými příspěvky studenta a pedagoga včetně přiložených souborů. Celkový popis uživatelského rozhraní modulu student a celého systému je uveden v uživatelské příručce.

a) Zobrazení výuky



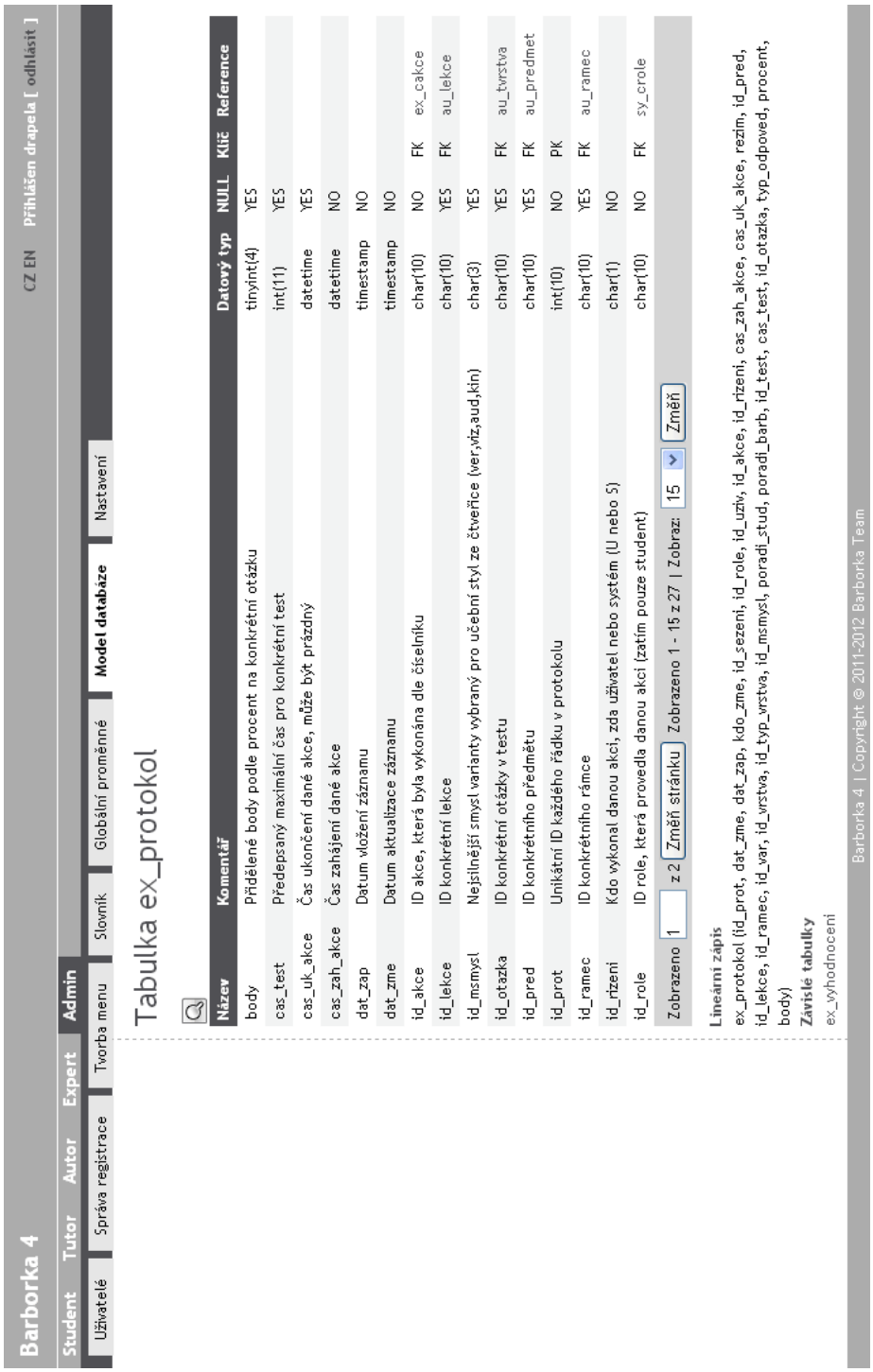
Obrázek 19: Ukázka výuky

b) Zobrazení autotestu



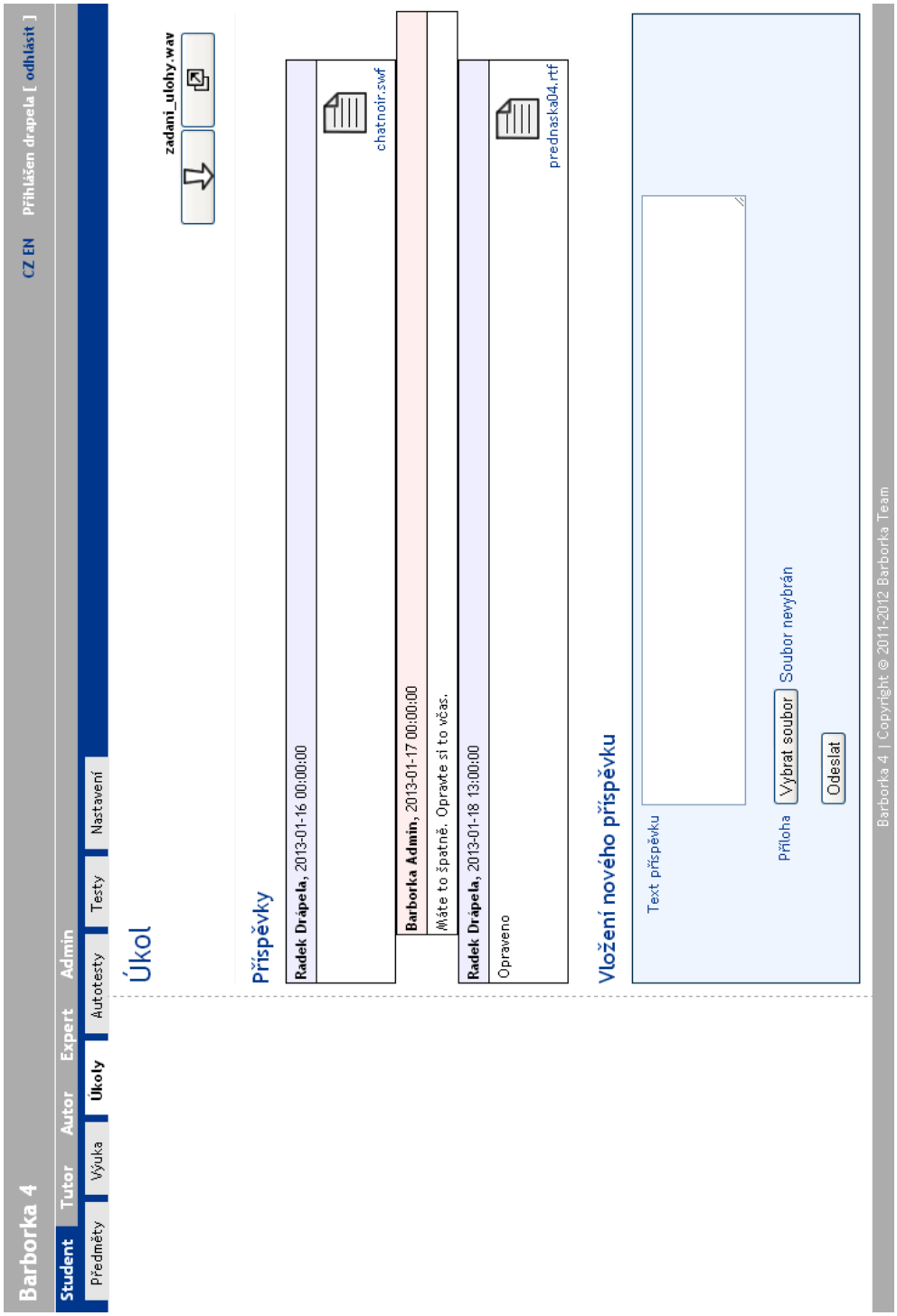
Obrázek 20: Ukázka testování

c) Zobrazení tabulky v modelu databáze



Obrázek 21: Ukázka modelu databáze

d) Zobrazení detailu úkolu



Obrázek 22: Ukázka úkolu

5 Závěr

V této diplomové práci byla popsána teorie adaptivního e-learningu, podle které byl navržen a vytvořen systém Barborka 4. Všechny části této teorie se promítly až do implementace, kde výsledný celek vytvořil požadovaný systém a výsledné řízení výuky adaptivně reaguje na charakteristiky studenta. Tímto byla hlavní část projektu úspěšně dokončena a byl vytvořen plnohodnotný systém řízení výuky. Dosud naimplementované funkcionality byly analyzovány a programovány během 20 měsíců. Byla provedena datová i funkční analýza a návrh nejdůležitějších částí systému, zaměřených na výuku a testování. Všechny elementární funkce uvedené v příloze jsou v systému úspěšně naimplementovány.

Jednotlivé schůze vývojového týmu probíhaly pravidelně každý týden (s výjimečnými přestávkami) a všechny vytvořené části systému, které se v této práci vyskytují, se důkladně společně s ostatními vývojáři konzultovaly, analyzovaly a pak byly všem prezentovány. Následně probíhala diskuze o možných úpravách, vylepšeních, či jiném možném využití. Také na přelomu května a června roku 2012 se konal závěrečný workshop k projektu Adaptivní individualizovaná výuka v e-learningu, na kterém byly představeny výsledky provedeného výzkumu z této oblasti.

Systém je z hlavní části naimplementován, obsahuje základní požadované funkce a je připraven pro použití v běžném provozu. Nové požadavky na systém, obzvlášť výuku, ovšem neustále přibývají, a tak jsou plánována nová rozšíření systému a výuky jako například: různé režimy výuky, vlastní poznámky studenta u zobrazených vrstev, historie odpovědí na otázky, výběr jiné otázky, nové speciální typy odpovědí atd. V současné době již funguje ostrá verze, kde je možné vytvářet první předměty a autorské opory. Jedním takovým je předmět Angličtina na Slezské univerzitě v Karviné. Pilotní výuka tohoto předmětu bude spuštěna již v květnu současného roku. Avšak až při úplném dokončení implementace je naplánováno systém plně otestovat a optimalizovat, jak na části zdrojového kódu, tak i na části databázového návrhu.

Předpokládá se, že systém Barborka 4 v současném období a blízké budoucnosti převezme z části roli pedagoga na kurzech a školách, které se rozhodly tento systém používat. Studenti se budou moci učit i testovat sami, dojde ke zkvalitnění výuky, jak z pohledu učitele, tak i z pohledu studenta a výsledky adaptivní výuky budou ověřeny v praxi.

6 Reference

[1] KOSTOLÁNYOVÁ, Kateřina. *Teorie adaptivního e-learningu.* Vyd. 1. Ostrava: Ostravská univerzita v Ostravě, 2012, 118 s. ISBN 978-80-7464-014-8.

[2] KAPOUNOVÁ, Jana. *Přístupy k evaluaci eLearningu.* Vyd. 1. Ostrava: Ostravská univerzita v Ostravě, 2012, 190 s. ISBN 978-80-7464-121-3.

[3] KORVINY, Petr. Moodle (nejen) na OPF: příručka uživatele. *MENDELU* [online]. 2005 [cit. 2013-04-27]. Dostupné z: https://akela.mendelu.cz/∼nadule/vyuka/IT/Moodle_OPF/

[4] BRDIČKA, Bořivoj. *Role internetu ve vzdělávání: studijní materiál pro učitele snažící se uplatnit moderní technologie ve výuce.* Kladno: Aisis, 2003, 122 s. ISBN 80-239-0106-0.

[5] ŠARMANOVÁ, Jana, Kateřina KOSTOLÁNYOVÁ. OSTRAVSKÁ UNIVERZITA V OSTRAVĚ. *Tvorba adaptivních e-learningových opor: metodická příručka pro autory.* Ostrava, 2011.

[6] ROUSE, Margaret. What is learning management system (LMS)?. TECHTARGET. *CIO information, news and tips* [online]. @2007-2013 [cit. 2013-04-27]. Dostupné z: http://searchcio.techtarget.com/definition/learning-management-system

[7] LMS. *Metodický portál RVP.CZ* [online]. 2010 [cit. 2013-04-27]. Dostupné z: http://wiki.rvp.cz/Knihovna/1.Pedagogicky_lexikon/L/LMS

[8] GAGNÉ, Robert M. *Podmínky učení.* 1. vyd. Praha: Stát. pedag. nakl., 1975, 287 s.

[9] PASCH, Martin. *Od vzdělávacího programu k vyučovací hodině.* Vyd. 2. Praha: Portál, 2005, 416 s. ISBN 80-736-7054-2.

[10] KOMENSKÝ, Jan Amos. *Didaktika velká.* Vyd. 3. Editor Josef Hendrich. Překlad Augustin Krejčí. Brno: Komenium, 1948, 252 s. Pedagogické klasobraní, sv. 2.

[11] KOSTOLÁNYOVÁ, Kateřina a Jana ŠARMANOVÁ. *Intelligent individualization of study through E-learning.* Ostrava: Ostravská univerzita, 2008. ISBN 978-80-7368-577-5.

[12] KOSTOLÁNYOVÁ, Kateřina, Jana ŠARMANOVÁ a Ondřej TAKÁCS. *Learning Characteristics: Proceedings of the 9th European Conference on e-Learning.* Instituto Superior de Engenharia do Porto:Academic Publishing Limited,2010. ISBN 978-1-906638-83-2.

[13] KOSTOLÁNYOVÁ, Kateřina a Jana ŠARMANOVÁ. *Theoretical and Practical Aspects of Distance Learning.* Methodology for Creating Adaptive Teaching Support. Katowice: Studio NOA, 2009a. ISBN 83-600-7130-6.

[14] ŠARMANOVÁ, Jana. *Metodika tvorby adaptivních e-learningových učebnic.* 1. vyd. Ostrava, 2011, 30 s.

[15] MAREŠ, Jiří. *Styly učení žáků a studentů.* 1. vyd. Praha: Portál, 1998, 239 s. ISBN 80-717-8246-7.

[16] ŠARMANOVÁ, Jana. VYSOKÁ ŠKOLA BÁŇSKÁ TECHNICKÁ UNIVERZITA OSTRAVA. *Personalizace výuky prostřednictvím e-learningu* [online]. 2009-2012 [cit. 2013-04-27]. Dostupné z: http://www.person.vsb.cz/

[17] KOSTOLÁNYOVÁ, Kateřina. OSTRAVSKÁ UNIVERZITA V OSTRAVĚ. *Adaptivní individualizovaná výuka v e-learningu* [online]. 2009-2012 [cit. 2013-04-27]. Dostupné z: http://projekty.osu.cz/adaptivita/

[18] DRÁBEK, Tomáš. *Informační subsystém evidence studia pro LMS Barborka.* Ostrava, 2007. Diplomová práce. Vysoká škola báňská - Technická univerzita Ostrava. Fakulta elektrotechniky a informatiky. Vedoucí práce Jana Šarmanová.

[19] HOLUB, Libor. *Automatizované řízení adaptivní výuky v e-learningu podle stylů učení studenta.* Ostrava, 2010. Disertační práce. Vysoká škola báňská - Technická univerzita Ostrava. Fakulta elektrotechniky a informatiky. Vedoucí práce Jana Šarmanová.

[20] FASUGA, Radoslav. *Inteligentní metody ve vzdělávání: Model virtuálního výukového prostředí, znalostní sítě, profilu studenta a popis základních metod virtuálního učitele a tutora.* Ostrava, 2009. Disertační práce. Vysoká škola báňská - Technická univerzita Ostrava. Fakulta elektrotechniky a informatiky. Vedoucí práce Jana Šarmanová.

[21] PHP Licensing. *PHP: Hypertext Preprocessor* [online]. @2001-2013 [cit. 2013-04-27]. Dostupné z: http://php.net/license/index.php

[22] David Grudl – rozhovor - Rozhovory s internetovými osobnostmi. PĚSTUJEME WEB. *Rozhovory s internetovými osobnostmi* [online]. @2008-2009 [cit. 2013-04-27]. Dostupné z: http://www.pestujemeweb.cz/clanky-rozhovory/text-27-david-grudl-rozhovor.html

[23] NETTE FOUNDATION. *Rychlý a pohodlný vývoj webových aplikací v PHP — Nette Framework* [online]. @2008, 2013 [cit. 2013-04-27]. Dostupné z: http://nette.org/cs/

[24] Licenční politika — Nette Framework. NETTE FOUNDATION. *Rychlý a pohodlný vývoj webových aplikací v PHP — Nette Framework* [online]. 2008, 2013 [cit. 2013-04-27]. Dostupné z: http://nette.org/cs/license

[25] PHP Nette Framework - plugin detail. ORACLE CORPORATION AND/OR ITS AFFILIATES. *NetBeans IDE: The Smarter and Faster Way to Code* [online]. @2012, 2013 [cit. 2013-04-27]. Dostupné z: http://plugins.netbeans.org/plugin/32720/

[26] Databáze & ORM — Nette Framework. NETTE FOUNDATION. *Rychlý a pohodlný vývoj webových aplikací v PHP — Nette Framework* [online]. @2008, 2013 [cit. 2013-04-27]. Dostupné z: http://doc.nette.org/cs/database#toc-prace-s-tabulkami

[27] Zabezpečení před zranitelnostmi — Nette Framework. NETTE FOUNDATION. *Rychlý a pohodlný vývoj webových aplikací v PHP — Nette Framework* [online]. @2008, 2013 [cit. 2013-04-27]. Dostupné z: http://doc.nette.org/cs/vulnerability-protection

[28] MVC aplikace & presentery — Nette Framework. NETTE FOUNDATION. *Rychlý a pohodlný vývoj webových aplikací v PHP — Nette Framework* [online]. @2008, 2013 [cit. 2013-04-27]. Dostupné z: http://doc.nette.org/cs/presenters#toc-model-view-controller-mvc

A Seznam elementárních funkcí z požadavků

Modul Public
Pu_01_Prihlaseni - přihlášení uživatele do systému
Pu_02_Registrace - registrace nového studenta do systému, pokud je registrace zpřístupněna, je potřeba znát heslo Barborky

Modul Student
St_01_Predmety
- **St_01_01_MePredmety**
  - **St_01_01_01_SeznamMychPredmetu** - zobrazení seznamu všech zapsaných předmětů studenta v daném akademickém roce
  - **St_01_01_02_DetailMehoPredmetu** - zobrazení informací o zapsaném předmětu studenta
  - **St_01_01_03_VyberMehoPredmetu** - zvolení daného předmětu, pro který se bude zobrazovat výuka a filtrovat autotesty, testy i úkoly
- **St_01_02_VerejnePredmety**
  - **St_01_02_01_SeznamVerejnychPredmetu** - zobrazení seznamu všech veřejným předmětů systému v daném akademickém roce
  - **St_01_02_02_DetailVerejnehoPredmetu** - zobrazení informací o veřejném předmětu studenta
  - **St_01_02_03_VyberVerejnehoPredmetu** - zvolení veřejného předmětu, pro který se bude zobrazovat výuka a filtrovat autotesty, testy i úkoly
- **St_01_03_PrihlaseniKPredmetu**
  - **St_01_03_01_SeznamDostupnychPredmetu** - zobrazení seznamu předmětů, které jsou zpřístupněné a do kterých se student může s heslem přihlásit
  - **St_01_03_02_DetailDostupnehoPredmetu** - zobrazení informací o dostupném předmětu pro přihlášení
  - **St_01_03_03_Prihlaseni** - přihlášení studenta k předmětu v daném akadamickém roce, je potřeba znát heslo

St_02_Vyuka
- **St_02_01_ZobrazeníVrstev** - zobrazení všech typů existujících vrstev obsahující komponenty, u testovacích otázek se zobrazují formuláře na odpovědi a reakce virtuálního učitele.
- **St_02_02_ZobrazeníDostupnýchVariant** - varianty rámce se zobrazují v 12-tici prvků (4 smyslové varianty ve 3 hloubkách)
- **St_02_03_ZobrazeníNavigacePredmetu** - použití generátoru menu pro zobrazení hierarchické struktury předmětu
- **St_02_04_ZobrazeníTlacitek** - zobrazení prvků pro resetování aktuálního rámce, pokračování ve výuce, přechod na předcházející rámec a na další rámec
- **St_02_05_VyberPredmetu** - výběr předmětu z navigace, pro který se bude zobrazovat výuka a filtrovat autotesty, testy i úkoly
- **St_02_06_VyberRamce** - zvolení rámce, ze kterého se zobrazí příslušné vrstvy
- **St_02_07_VyberVarianty** - zvolení jiné varianty pro aktuální zvolenou vrstvu

St_02_08_VyberVrstvy - výběr jiné již zobrazené vrstvy
St_02_09_PokracovaniVeVyuce - přechod na následující vrstvu v aktuálním rámci, pokud se jedná o poslední vrstvu, zobrazí se první vrstva následujícího rámce
St_02_10_ResetRamce - převedení aktuálního rámce do výchozího stavu a zobrazení pouze první vrstvy
St_02_11_OdeslaniOdpovedi - pro testovací otázku se odešle formulář s odpovědí, vyhodnotí se a vypíše se reakce

St_03_Ukoly
St_03_01_SeznamUkolu - seznam všech úkolů dostupných nebo filtrovaných podle vybraného předmětu
St_03_02_DetailUkolu
St_03_02_01_ZobrazeniZadani - zobrazení zadání úkolu pomocí komponent
St_03_02_02_ZobrazeniPrispevku - zobrazení příspěvků v diskuzi úkolu
St_03_02_03_VlozeniPrispevku - vložení příspěvku do diskuze a upload souboru

St_04_Autotesty
St_04_01_SeznamAktivnichAutotestu - zobrazení aktivních autotestů, které jsou dostupné k okamžitému spuštění
St_04_02_SpusteniAutotestu - vygenerování náhodných otázek a odpovědí
St_04_03_ZobrazeniPokynu - zobrazení komponent obsahující pokyny k vypracování autotestu
St_04_04_MujAutotest - zobrazení autotestu v původním okně se všemi otázkami
St_04_04_01_ZobrazeniAutotestu - zobrazení všech vygenerovaných otázek (obsahující komponenty), formuláře pro odpovědi, hlavičky a časomíry
St_04_04_02_UlozeniAutotestu - uložení dočasných studentem vyplněných odpovědí spuštěného autotestu
St_04_04_03_UkonceniAutotestu - ukončení autotestu a odeslání formuláře s odpověďmi pro vyhodnocení
St_04_05_MujAutotestOkno - zobrazení autotestu v novém okně po 1 otázce
St_04_05_01_PredchoziOtazka - uložení otázky a přechod na předchozí otázku aktuálního autotestu
St_04_05_02_DalsiOtazka - uložení otázky a přechod na následující otázku aktuálního autotestu
St_04_05_03_ZobrazeniAutotestuOkno - zobrazení dané otázky autotestu (obsahující komponenty), formuláře s odpověďmi, hlavičky a časomíry
St_04_05_04_UkonceniAutotestuOkno - ukončení autotestu a odeslání formuláře pro vyhodnocení odpovědí
St_04_06_ZobrazeniVyhodnoceni - zobrazení hlavičky autotestu s výsledkem a vyhodnocení jednotlivých otázek (tvořící komponenty)

St_05_Testy
St_05_01_PrihlaseniKTerminu
St_05_01_01_SeznamDostupnychTerminu - zobrazení seznamu termínů testů ze zapsaných předmětů studenta, ke kterým se může přihlásit

St_05_01_02_Prihlaseni - přihlášení studenta k testu daného termínu

St_05_02_MojeTesty

St_05_02_01_SeznamMychTestu - zobrazení seznamu se všemi testy uživatele - vykonané, aktuální, budoucí

St_05_02_02_DetailTestu

St_05_02_02_01_ZobrazeniDetailu - zobrazení hlavičky obsahující dostupné informace o daném testu

St_05_02_02_02_ZobrazeniVyhodnoceni - zobrazení výsledky s vyhodnocením jednotlivých otázek, pokud je zobrazení zveřejněno

St_05_02_02_03_OdhlaseniZTerminu - odhlášení studenta z termínu daného testu dříve, než je test spuštěn

St_05_03_SeznamAktivnichTestu - zobrazení aktivních testů, které jsou dostupné k okamžitému spuštění

St_05_04_SpusteniTestu - výběr daných otázek a odpovědí ze předem vygenerované struktury testů

St_05_05_ZobrazeniPokynu - zobrazení komponent obsahující pokyny k vypracování testu

St_05_06_MujTest - zobrazení testu v původním okně se všemi otázkami testu

St_05_06_01_ZobrazeniTestu - zobrazení všech vybraných otázek (obsahující komponenty), formuláře pro odpovědi, hlavičky a časomíry

St_05_06_02_UlozeniTestu - uložení dočasných studentem vyplněných odpovědí spuštěného testu

St_05_06_03_UkonceniTestu - ukončení testu a odeslání formuláře s odpověďmi pro vyhodnocení

St_05_05_MujTestOkno - zobrazení testu v novém okně po 1 otázce

St_05_05_01_PředchoziOtazka - uložení otázky a přechod na předchozí otázku aktuálního testu

St_05_05_02_DalsiOtazka - uložení otázky a přechod na následující otázku aktuálního testu

St_05_05_03_ZobrazeniTestuOkno - zobrazení dané otázky testu (obsahující komponenty), formuláře s odpověďmi, hlavičky a časomíry

St_05_05_04_UkonceniTestuOkno - ukončení testu a odeslání formuláře pro vyhodnocení odpovědí

St_05_06_ZobrazeniVyhodnoceni - zobrazení hlavičky testu s výsledkem a vyhodnocení jednotlivých otázek (tvořící komponenty)

St_06_Nastaveni

St_06_01_ZmenaUdaju - zobrazení formuláře s povolenými údaji studenta a akce pro změnu těchto údajů

St_06_02_ZmenaHesla - zobrazení formuláře pro změnu hesla studenta a akce pro změnu hesla

Modul Administrátor/systém

Sy_01_Uzivatele

Sy_01_01_SeznamUzivatelu - zobrazení seznamu všech uživatelů systému

Sy_01_02_DetailUzivatele - zobrazení neživého formuláře obsahujícího všechny údaje o zvoleném uživateli a aktualizace údajů
Sy_01_03_NovyUzivatel - zobrazení živého formuláře pro vytvoření nového uživatele
Sy_01_04_UpravaUzivatele - zobrazení živého formuláře obsahující všechny údaje o zvoleném uživateli s možností úpravy
Sy_01_05_OdstraneniUzivatele - odstranění uživatele ze systému
Sy_01_06_ImportUzivatelu - hromadný import uživatelů do systému ze souboru CSV
Sy_01_07_VlozeniUzivatele - samotné vložení uživatele do systému

Sy_02_SpravaRegistrace
Sy_02_01_SeznamAktualnichZpristupneniRegistrace - zobrazení seznamu všech aktuální zpřístupnění registrace uživatelů do systému
Sy_02_02_NoveZpristupneniRegistrace - zobrazení živého formuláře a vložení nového zpřístupění registrace
Sy_02_03_UpravaZpristupneniRegistrace - zobrazení živého formuláře obsahujícího všechny údaje o zpřístupnění a následná aktualizace údajů
Sy_02_04_UkonceniZpristupneniRegistrace - nastavení konce zvoleného zpřístupnění na aktuální čas
Sy_02_05_HistorieZpristupneni - zobrazení všech zpřístupnění, jež skončila

Sy_03_TvorbaMenu
Sy_03_01_DetailPolozky - zobrazení seznamu podpoložek zvolené položky menu
Sy_03_02_NovyPolozka - zobrazení živého formuláře a vložení položky menu
Sy_03_03_UpravaPolozky - zobrazení živého formuláře obsahujícího všechny údaje o položce a aktualizace údajů
Sy_03_04_OdstraneniPolozky - odstranění položky menu

Sy_04_Slovnik
Sy_04_01_SlovniSpojeni
Sy_04_01_01_SeznamSlovnichSpojeni - zobrazení seznamu s frázemi (slovními spojeními) ve slovníku
Sy_04_01_02_NovyVyraz - zobrazení živého formuláře a vložení nové fráze do slovníku
Sy_04_01_03_UpravaVyrazu - zobrazení živého formuláře obsahujícího výraz s překladem a aktualizace údajů
Sy_04_01_04_OdstraneniVyrazu - odstranění výrazu ze slovníku
Sy_04_02_TypySlovnichSpojeni
Sy_04_02_01_SeznamTypu - zobrazení seznamu s typy frází (typy slovních spojení) použitých ve slovníku
Sy_04_02_02_NovyTyp - zobrazení živého formuláře a vložení nového typu
Sy_04_02_03_UpravaTypu - zobrazení živého formuláře s popisem a aktualizace typu
Sy_04_02_04_OdstraneniTypu - odstranění typu

Sy_05_GlobalniPromenne

Sy_05_01_SeznamPromennych - zobrazení všech globálních proměnných systému
Sy_05_02_NovaPromenna - zobrazení živého formuláře a vložení globální proměnné
Sy_05_03_UpravaPromenne - zobrazení živého formuláře s hodnotou a aktualizace typu
Sy_05_04_OdstraneniPromenna - odstranění globální proměnné

Sy_06_ModelDatabaze
Sy_06_01_SeznamTabulek - zobrazení seznamu databázových tabulek systému
Sy_06_02_DetailTabulky - zobrazení atributů, lineárního zápisu a závislých tabulek zvolené tabulky

Sy_07_Nastaveni
Sy_07_01_ZmenaUdaju - změna všech údajů administrátora
Sy_07_02_ZmenaHesla - změna hesla administrátora

Sy_08_ZobrazeniUI
Sy_08_01_ZobrazeniMenu - generování menu podle rolí a jemných práv aktuálního uživatele
Sy_08_01_01_GenerovaniHlavnihoMenu - generování hlavního menu (menu s moduly podle rolí a submenu - 0. a 1. úroveň DFD)
Sy_08_01_02_GenerovaniLevehoMenu - generování rozklikávacího levého menu (2. a další úroveň DFD)
Sy_08_02_ZobrazeniJazyku - zobrazení dostupných jazyků v systému
Sy_08_03_ZobrazeniSpolecnychKomponent - zobrazení různých systémových komponent společné pro elementární funkce

Část modulu Expert

Ex_04_Studenti
Ex_04_01_SeznamStudentů - zobrazení všech studentů v systému (reálných i virtuálních)
Ex_04_02_NovyVirtualniStudent - zobrazení živého formuláře pro učební styl a vložení nového virtuálního studenta
Ex_04_03_UpravaStudenta - zobrazení živého formuláře s údaji o učebním stylu studenta a aktualizace těchto údajů
Ex_04_04_OdstraneniVirtualnihoStudenta - odstranění virtuálního studenta

Ex_05_ModelovaniVyuky
Ex_05_01_ModelVyuky
Ex_05_01_01_VyberStudentu - zobrazení seznamu všech studentů a hromadný výběr studentů, pro které se bude provádět modelování
Ex_05_01_02_VyberOpory - výběr virtuální opory a nastavení parametrů pro generování grafu
Ex_05_01_03_ZobrazeniGrafu - zobrazení grafu modelu výuky

Ex_06_Nastaveni
Ex_06_01_ZmenaUdaju - zobrazení fomuláře s povolenými údaji experta a akce pro změnu těchto údajů
Ex_06_02_ZmenaHesla - zobrazení fomuláře pro změnu hesla experta a změna hesla

B Datový slovník

Společné atributy pro všechny tabulky kromě sy_filtry

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
kdo_zme	ID uživatele, který provedl poslední aktualizaci záznamu	char	10	Ne		Ne		Uziv999999
dat_zap	Datum vložení záznamu	timestamp		Ne		Ne	1	
dat_zme	Datum aktualizace záznamu	timestamp		Ne		Ne	1	

Tabulka 6: Datový slovník: Společné atributy

Tabulka st_ntvrstva
Nastavení testovacích vrstev

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_ntvr	Identifikátor jednoho nastavení testovací vrstvy.	char	10	Ne	primární	Ano		Ntvr999999
id_tvrstva	Identifikátor test vrstvy	char	10	Ne	cizí (au_tvrstva)	Ano		Tvrs999999
id_uziv	Identifikátor uživatele.	char	10	Ne	cizí (sy_uziv)	Ano		Uziv999999
odpoved	Text odpovědi nebo identifikátory variantních odpovědí.	text		Ne		Ne	2	
rezim	Zdali se jedná o výuku, autotest či o test	char	1	Ne		Ano		V,A,T
stav	Označení stavu otázky od experta	int		Ne		Ne		
zprava	Text reakce k odpovědi	text		Ano		Ne		

Tabulka 7: Datový slovník: Tabulka st_ntvrstva

Tabulka st_student
Studenti

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_uziv	Identifikátor uživatele	char	10	Ne	primární, cizí (sy_uziv)	Ano		Uziv999999
dat_nar	Datum narození studenta	date		Ano		Ne		
email	Kontaktní email studenta	varchar	80	Ne		Ne	3	
id_oboru	Studijní obor studenta	char	10	Ano	cizí (tu_obor)	Ano		Obor999999
jmeno	Jméno studenta	varchar	50	Ne		Ne		
prijmeni	Příjmení studenta	varchar	80	Ne		Ne		
rocnik	Aktuální ročník studenta	int		Ano		Ne		
titul	Tituly studenta	varchar	20	Ano		Ne	4	

virtualni	Binární atribut určující, zdali se jedná o virtuálního studenta	tinyint		Ne		Ne		0,1

Tabulka 8: Datový slovník: Tabulka st_student

Tabulka st_ustyl
Typické vlastnosti studenta, učební styl

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_uziv	Identifikátor uživatele	char	10	Ne	primární, cizí (st_student)	Ano		Uziv999999
saud	Auditivní smyslové vnímání	int		Ano		Ne		0 až 100
skin	Kinestetické smyslové vnímání	int		Ano		Ne		0 až 100
stafek	Afektivní aspekty	int		Ano		Ne		-100 až 100
stareg	Autoregulace	int		Ano		Ne		-100 až 100
stdetail	Postup učení - detailistický	int		Ano		Ne		0 až 100
stexp	Způsob zpracování informací - experimentování	int		Ano		Ne		0 až 100
sthol	Postup učení - holistický	int		Ano		Ne		0 až 100
stpoj	Pojetí učení	int		Ano		Ne		-100 až 100
stsoc	Sociální aspekty	int		Ano		Ne		0 až 100
stsyst	Systematičnost	int		Ano		Ne		0 až 100
stteor	Způsob zpracování informací - teoretické odvozování	int		Ano		Ne		0 až 100

Tabulka 9: Datový slovník: Tabulka st_ustyl

Tabulka sy_cmodul
Číselník pro moduly

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_cmod	Identifikátor názvu modulu v číselníku	char	10	Ne	primární	Ano		CMod999999
nazev_m	Název modulu	varchar	10	Ne		Ne		Admin, Autor, Expert, Student, Tutor, Ostatni, Public

Tabulka 10: Datový slovník: Tabulka sy_cmodul

Tabulka sy_crole
Číselník práv pro jednotlivé role

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_crol	Identifikátor role	char	10	Ne	primární	Ano		CRol999999
nazev_r	Název role	varchar	10	Ne		Ne		admin, autor, expert, student, tutor
prava_r	Zřetězená kolekce názvů modulů, ke kterým má uživatel přístup. Názvy jsou oddělené čárkou.	text		Ne		Ne		St, Tu, Au, Ex, Sy

Tabulka 11: Datový slovník: Tabulka sy_crole

Tabulka sy_ctext
Číselník typů textů ve slovníku

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_ctex	Identifikátor typu textu	char	10	Ne	primární	Ano		CTex999999
popis_typ	Popis typu textu ve slovníku	text		Ne		Ne		
zkr_typ	Zkratka typu textu pro zobrazení	varchar	50	Ne		Ano		

Tabulka 12: Datový slovník: Tabulka sy_ctext

Tabulka sy_filtry
Předpočítané filtry pro zvýšení rychlosti jejich výpočtu

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_filt	md5 otisk zadání filtru: tabulky odkud a kam a parametrů vyhledávání filtru	char	32	Ne	primární	Ano		MD5 hash
filtr	Parametry filtru pro sestavení dotazu	text		Ano		Ne		

Tabulka 13: Datový slovník: Tabulka sy_filtry

Tabulka sy_menu
Položka menu (elementární funkce) a příslušný odkaz

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_menu	Systémový identifikátor uzlu	char	10	Ne	primární	Ano		Menu999999
id_nadm	Systémový identifikátor rodičovského uzlu	char	10	Ano	cizí (sy_menu)	Ano		Menu999999
help_m	Nápověda k uzlu menu	text		Ano		Ne		
iden_m	Identifikátor elementární funkce	varchar	60	Ne		Ano	5	

nazev_m	Zobrazovaný název uzlu	text		Ne		Ne		
popis_m	Popis uzlu menu	text		Ano		Ne		
poradi_m	Pořadí uzlu mezi potomky rodičovského uzlu	int		Ne		Ne		
volani_m	Odkaz, který se volá pro zvolení uzlu menu	text		Ne		Ano		
zobrazit_m	Příznak, zdali se má položka v menu zobrazit	tinyint		Ne		Ano		0, 1

Tabulka 14: Datový slovník: Tabulka sy_menu

Tabulka sy_nastaveni
Globální proměnné systému

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_nast	Identifikátor globální proměnné.	char	10	Ne	primární	Ano		Nast999999
hodnota_n	Hodnota globální proměnné	text		Ne		Ne		
id_cmod	Identifikátor modulu v číselníku	char	10	Ano	cizí (sy_cmodul)	Ano		CMod999999
nazev_n	Název globální proměnné	varchar	50	Ne		Ano		

Tabulka 15: Datový slovník: Tabulka sy_nastaveni

Tabulka sy_slovnik
Slovník pro jazykové verze

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_slov	Systémový identifikátor slovního spojení ve slovníku	char	10	Ne	primární	Ano		Slov999999
id_ctex	Systémový identifikátor typu textu ve slovníku	char	10	Ne	cizí (sy_ctext)	Ano		CTex999999
iden_s	Identifikátor/klíč slovního spojení pro použití	varchar	200	Ne		Ano		
nazev_cz	Český výraz slovního spojení	text		Ano		Ne		
nazev_en	Anglický výraz slovního spojení	text		Ne		Ne		

Tabulka 16: Datový slovník: Tabulka sy_slovnik

Tabulka sy_soubor
Nahrané soubory do systému

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_soub	Systémový identifikátor uploadovaného souboru	char	10	Ne	primární	Ano		Soub999999

cesta	Relativní cesta k souboru z kořenového adresáře s uploadovanými soubory (bez názvu souboru)	varchar	200	Ano		Ne	6	
mime	MIME typ souboru (např. image/jpeg)	varchar	50	Ano		Ne	7	
nazev	Název souboru	varchar	100	Ne		Ne	8	

Tabulka 17: Datový slovník: Tabulka sy_soubor

Tabulka sy_unastaveni

Uživatelské systémové proměnné

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_unas	Identifikátor globální proměnné.	char	10	Ne	primární	Ano		Unas999999
hodnota_n	Hodnota globální proměnné	text		Ne		Ne		
id_uziv	Identifikátor uživatele.	char	10	Ne	cizí (sy_uziv)	Ano		Uziv999999
nazev_n	Název globální proměnné	varchar	50	Ne		Ano		

Tabulka 18: Datový slovník: Tabulka sy_unastaveni

Tabulka sy_uziv

Všichni užívatelé systému

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_uziv	Identifikátor uživatele	char	10	Ne	primární	Ano		Uziv999999
heslo	Vygenerovaný hash hesla uživatele	char	40	Ano		Ne		
hlavni_role	Číslo hlavní role	tinyint		Ne		Ne		1, 2, 3, 4, 5
ldap	Příznak pokud se bude uživatel přihlašovat z LDAP	tinyint		Ne		Ne		0, 1
login	Přihlašovací login uživatele	varchar	20	Ne		Ano		
prava	Zřetězená kolekce názvů modulů a funkcí, ke kterým má uživatel přístup.	text		Ne		Ne	9	
role_admin	Příznak role administrátora	tinyint		Ne		Ne		0, 1
role_autor	Příznak role autora	tinyint		Ne		Ne		0, 1
role_expert	Příznak role experta	tinyint		Ne		Ne		0, 1
role_student	Příznak role studenta	tinyint		Ne		Ne		0, 1
role_tutor	Příznak role tutora	tinyint		Ne		Ne		0, 1

Tabulka 19: Datový slovník: Tabulka sy_uziv

Tabulka sy_zamestnanec
Zaměstnanci škol, učitelé

Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_uziv	Identifikátor zaměstnance	char	10	Ne	primární, cizí (sy_uziv)	Ano		Uziv999999
dat_nar	Datum narození zaměstnance	date		Ano		Ne		
email	Kontaktní email zaměstnance	varchar	80	Ne		Ne	3	
id_katedry	Katedra zaměstnance	char	10	Ano	cizí (tu_katedra)	Ano		Kate999999
jmeno	Jméno zaměstnance	varchar	50	Ne		Ne		
prijmeni	Příjmení zaměstnance	varchar	80	Ne		Ne		
mistnost	Místnost zaměstnance	varchar	10	Ano		Ne		
telefon	Telefonní kontakt na zaměstnance	varchar	20	Ano		Ne		
titul	Tituly zaměstnance	varchar	20	Ano		Ne	4	

Tabulka 20: Datový slovník: Tabulka sy_zamestnanec

Tabulka sy_zpristupneni
Časové zpřístupnění registrace do systému pod heslem

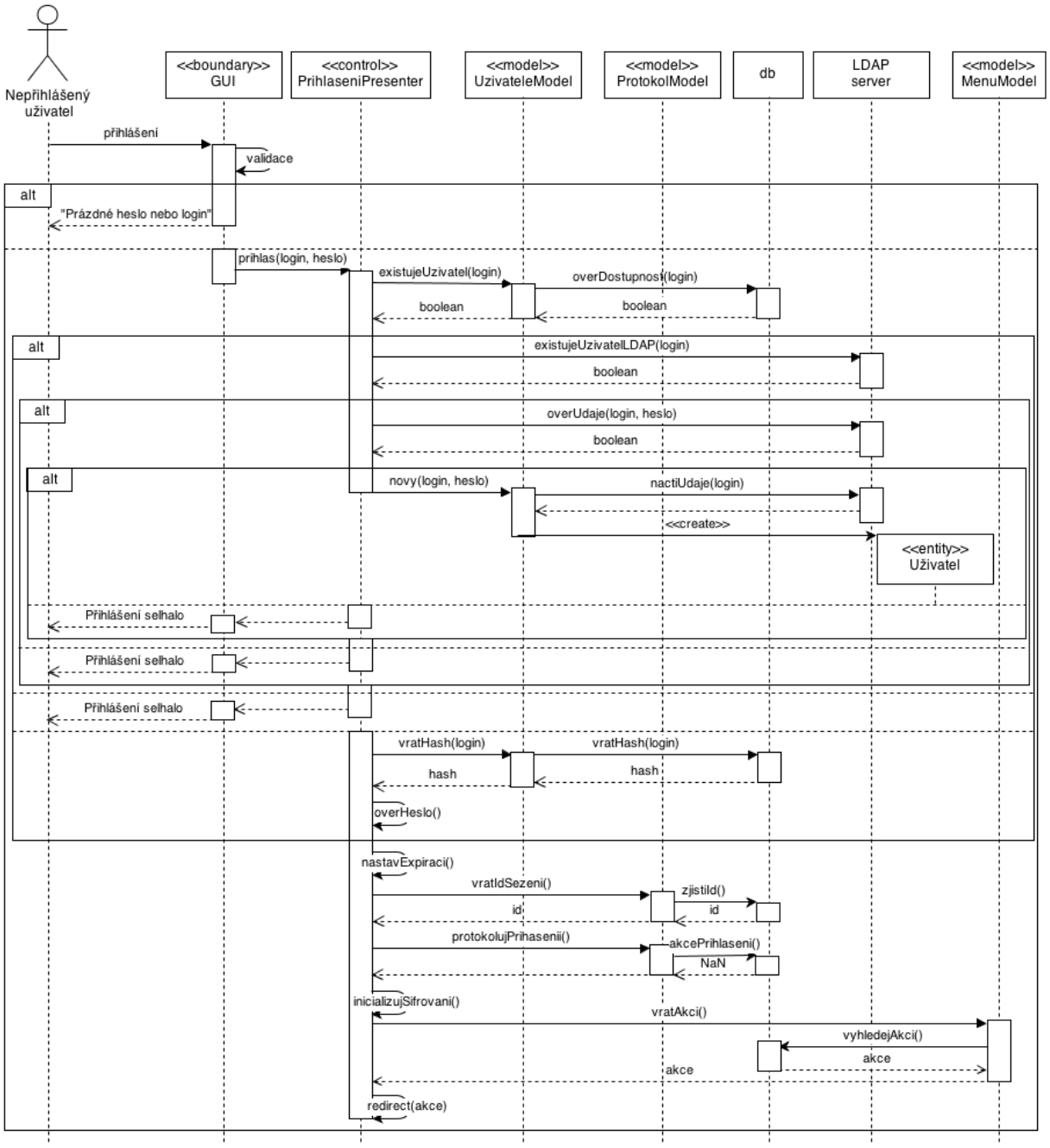
Název	Komentář	Dat. typ	Délka	Null	Klíč	Index	IO	Doména
id_zpri	Systémový identifikátor zpřístupnění registrace	char	10	Ne	primární	Ano		Zpri999999
heslo_z	Heslo k zpřístupnění registrace	varchar	30	Ne		Ne		
konec_z	Datum konce zpřístupnění registrace	datetime		Ne		Ano	10	
zacatek_z	Datum začátku zpřístupnění registrace	datetime		Ne		Ano	10	

Tabulka 21: Datový slovník: Tabulka sy_zpristupneni

Integritní omezení
1 dat_zap <= dat_zme
2 text pro tvořené odpovědi a „id_odp1|0,id_odp2|1,id_odp3|1,id_odp4|0”pro variantní
3 email má klasický tvar
4 tituly oddělené čárkou
5 prefix, dvojciferná čisla s pořadím a název elementární funkce oddělené podtržítky
6 cesta k souboru ve tvaru XX/XX, kde XX jsou 2 ciferné názvy složek
7 klasický mime typ ve tvaru obsahující lomítko „/”
8 klasický název souboru ve tvaru: jméno.přípona
9 zkratky modulů a identifikátory elementárních funkcí oddělené čárkou; u každého id elementární funkce může byt definována akce oddělena dvojtečkou
10 zacatek_z < konec_z

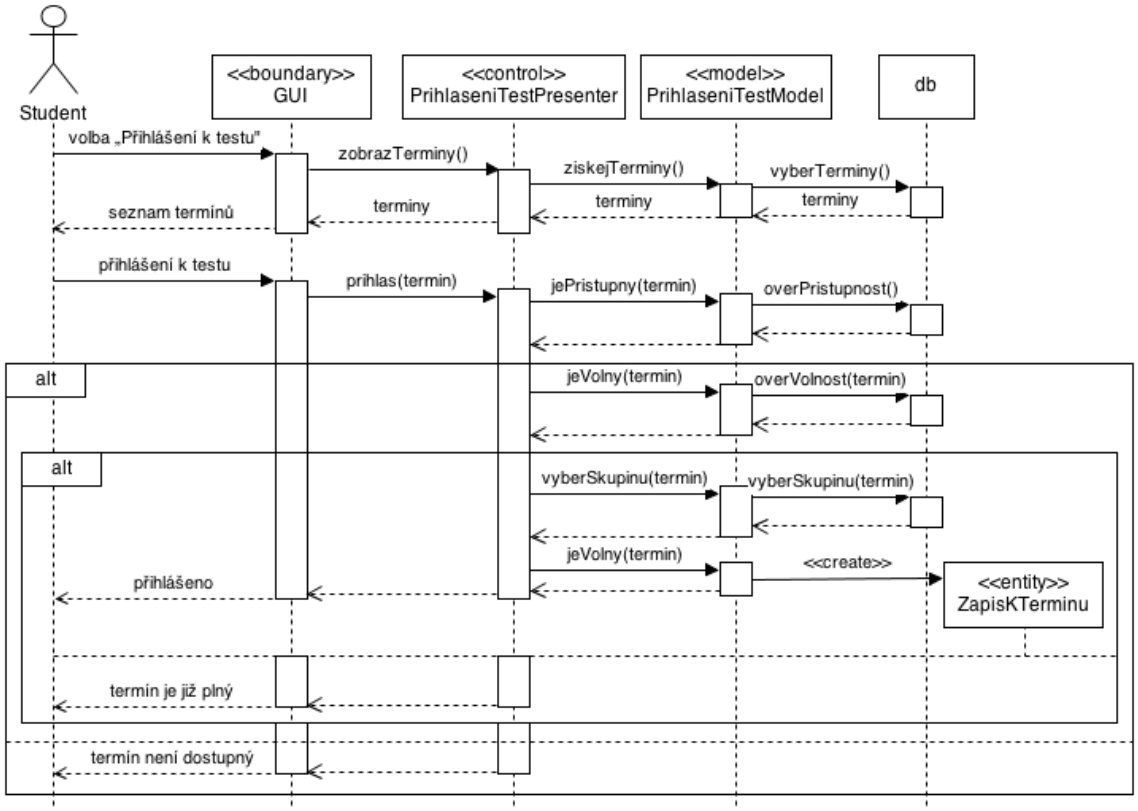
C Sekvenční diagramy

a) Přihlášení uživatele do systému (obrázek 23)



Obrázek 23: Sekvenční diagram: Přihlášení uživatele do systému

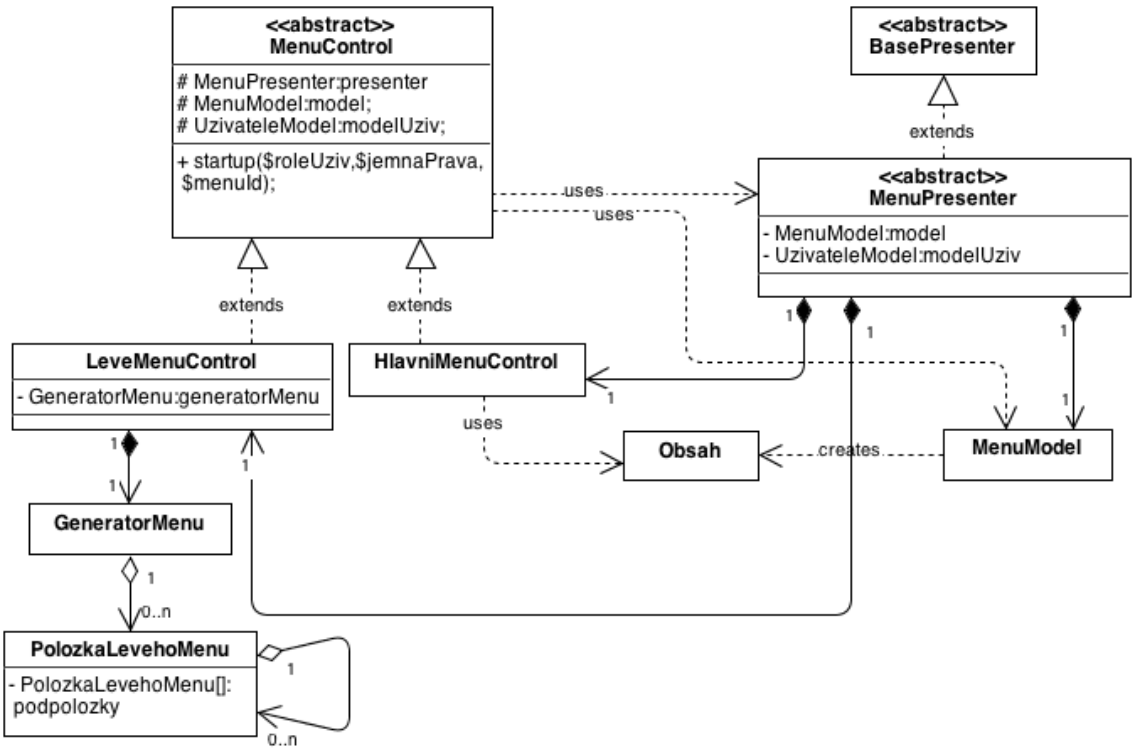
b) Přihlášení studenta k termínu testu (obrázek 24)



Obrázek 24: Sekvenční diagram: Přihlášení studenta k termínu testu

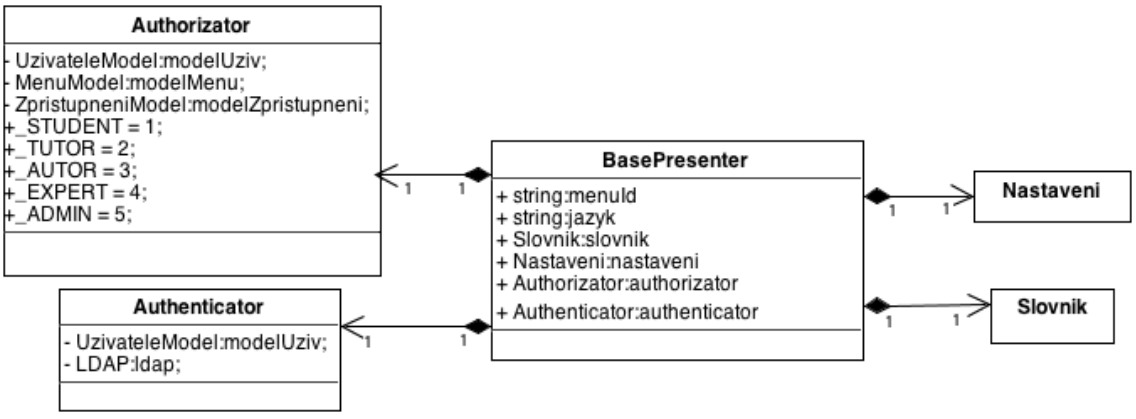
D Třídní diagramy

a) Menu (obrázek 25)



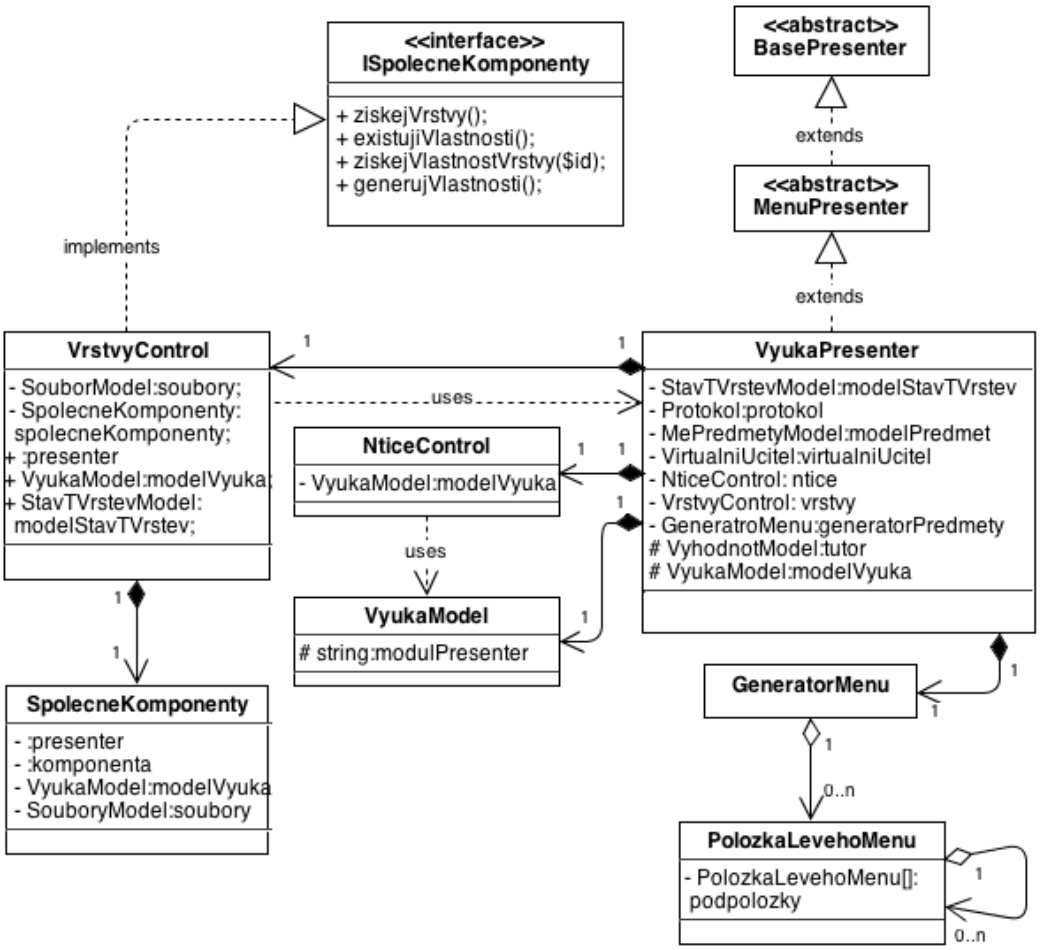
Obrázek 25: Třídní diagram: Menu

b) BasePresenter (obrázek 26)



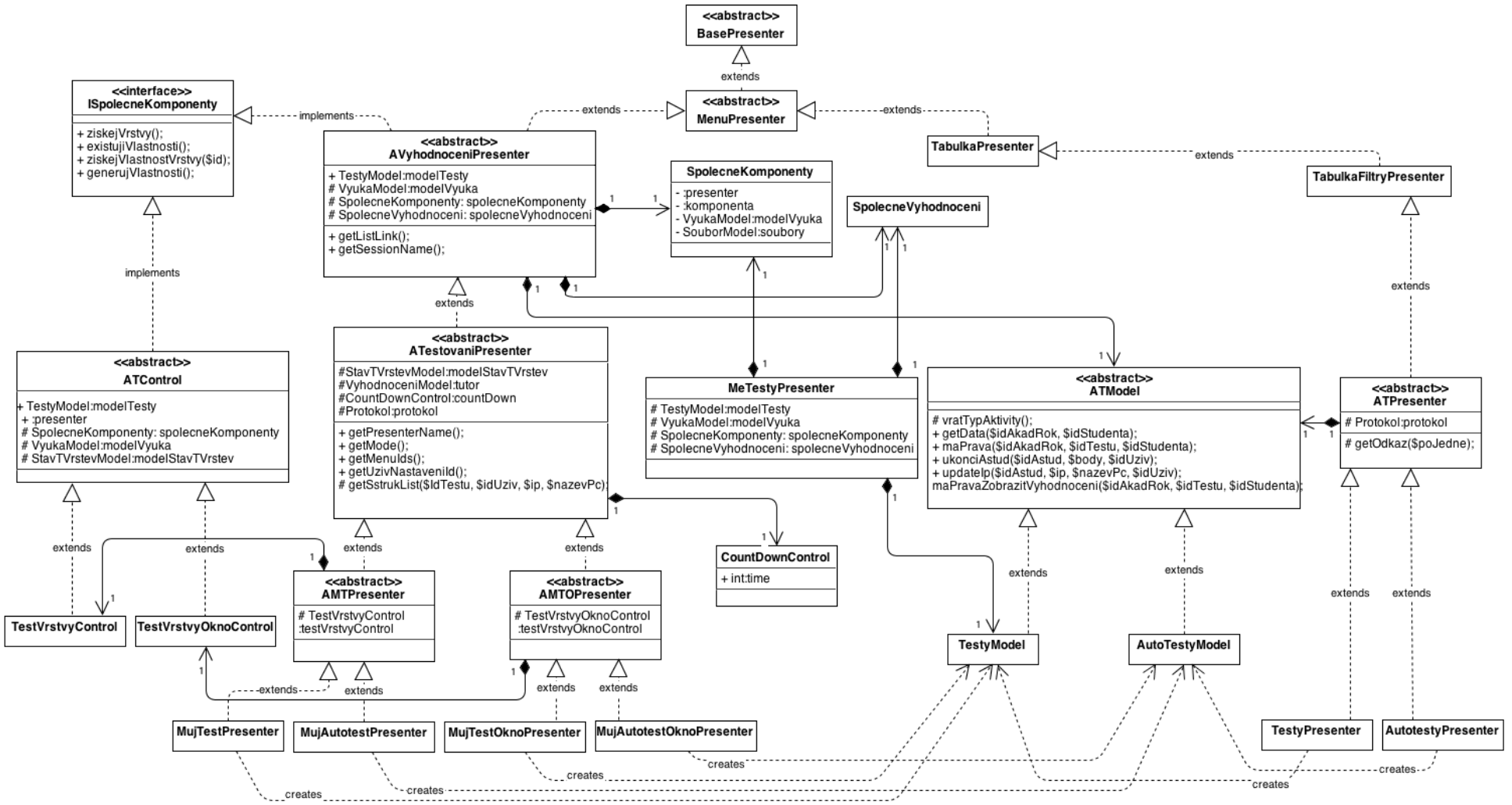
Obrázek 26: Třídní diagram: BasePresenter

c) Výuka (obrázek 27)



Obrázek 27: Třídní diagram: Výuka

d) Testování (obrázek 28)



Obrázek 28: Třídní diagram: Testování

E Certifikát účastníka

Obrázek 29: Certifikát účastníka za školení v oblasti adaptivní výuky

F Obsah přiloženého disku

Diplomová práce - Radek Drápela.pdf
Samotné vypracování diplomové práce, tedy tento aktuální dokument.
Zadání.pdf
Podepsané naskenované zadání diplomové práce.
Abstrakt.pdf
Český a anglický abstrakt této práce včetně klíčových slov ve zvláštním souboru.
Barborka 4 - Uživatelská příručka.pdf
Uživatelská příručka pro studenty a administrátory LMS Barborka 4.
Barborka 4 - Programátorská příručka.pdf
Programátorská příručka pro vývojáře LMS Barborka 4.